

Deep Neural Networks Inspired by Differential Equations

YONGSHUAI LIU, LIANFANG WANG, KUILIN QIN, QINGHUA ZHANG, FAQIANG WANG, LI

CUI, JUN LIU, and YUPING DUAN*, School of Mathematical Sciences, Beijing Normal University, China

TIEYONG ZENG, Department of Mathematics, The Chinese University of Hong Kong, Hong Kong

Deep learning has become a pivotal technology in fields such as computer vision, scientific computing, and dynamical systems, significantly advancing these disciplines. However, neural Networks persistently face challenges related to theoretical understanding, interpretability, and generalization. To address these issues, researchers are increasingly adopting a differential equations perspective to propose a unified theoretical framework and systematic design methodologies for neural networks. In this paper, we provide an extensive review of deep neural network architectures and dynamic modeling methods inspired by differential equations. We specifically examine deep neural network models and deterministic dynamical network constructs based on ordinary differential equations (ODEs), as well as regularization techniques and stochastic dynamical network models informed by stochastic differential equations (SDEs). We present numerical comparisons of these models to illustrate their characteristics and performance. Finally, we explore promising research directions in integrating differential equations with deep learning to offer new insights for developing intelligent computational methods that boast enhanced interpretability and generalization capabilities.

CCS Concepts: • Computing methodologies → Philosophical/theoretical foundations of artificial intelligence; • Networks → Network design principles; • Mathematics of computing → Differential equations.

Additional Key Words and Phrases: Deep neural networks, ordinary differential equation, stochastic differential equation, network structure, dynamic process

ACM Reference Format:

Yongshuai Liu, Lianfang Wang, Kuilin Qin, Qinghua Zhang, Faqiang Wang, Li Cui, Jun Liu, Yuping Duan, and Tieyong Zeng. 2018. Deep Neural Networks Inspired by Differential Equations. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation email (Conference acronym 'XX)*. ACM, New York, NY, USA, 35 pages. <https://doi.org/XXXXXXX.XXXXXXX>

1 Introduction

Deep learning has emerged as a transformative paradigm in modern artificial intelligence, driven by its remarkable success in tasks such as computer vision [10, 25, 250], natural language processing [38, 49, 195], autonomous driving [8, 81, 182] and medical measurement [2, 223]. These achievements are largely attributed to the confluence of three key factors: the proliferation of large-scale annotated datasets [248, 275], the rapid advancement in computational hardware [27, 221, 236] (e.g., GPUs and TPUs), and the continuous evolution of neural network architectures [153, 244, 258].

*Corresponding author.

Authors' Contact Information: Yongshuai Liu, lys_bnu@mail.bnu.edu.cn; Lianfang Wang, 202431130071@mail.bnu.edu.cn; Kuilin Qin, 202321130113@mail.bnu.edu.cn; Qinghua Zhang, qhzhang@mail.bnu.edu.cn; Faqiang Wang, fqwang@bnu.edu.cn; Li Cui, licui@bnu.edu.cn; Jun Liu, jliu@bnu.edu.cn; Yuping Duan, doveduan@gmail.com, School of Mathematical Sciences, Beijing Normal University, Beijing, China; Tieyong Zeng, Department of Mathematics, The Chinese University of Hong Kong, Shatin, Hong Kong, zeng@math.cuhk.edu.hk.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2018 Copyright held by the owner/author(s). Publication rights licensed to ACM.

Manuscript submitted to ACM

Despite these empirical successes, deep learning still faces challenges in interpretability and generalization [13, 87, 201], which limit its applicability in domains requiring rigorous safety and reliability guarantees [53, 210].

ODEs and SDEs have emerged as powerful theoretical frameworks providing novel insights and mathematical tools for tackling the challenges of interpretability and generalization in deep learning. Pioneering work by E et al. [257] conceptualized deep neural networks as discrete dynamic systems, framing the training phase as an optimal control problem. This groundbreaking perspective catalyzed the development of network architectures grounded in deterministic differential equations (DDEs), encompassing ODEs and partial differential equations (PDEs), and spurring substantial advancements [29, 74, 164, 211]. Stability in ODE-based models is often achieved by enforcing Lipschitz or contraction conditions on the function f , which ensures reliable forward and backward propagation and mitigates gradient explosion or vanishing. Overall, existing research can be broadly categorized into two main directions: 1) leveraging differential equations (DEs), including ODEs, PDEs and SDEs, frameworks for the theoretical interpretation of deep networks [4, 12, 74, 103, 133, 238, 257, 262, 282]; 2) guiding the design of network architecture with DEs [57, 80, 91, 113, 126, 164, 166, 211, 256, 283].

The intrinsic interpretability of ODEs arises from their deterministic nature; given an initial state, systems evolve along a single, predictable trajectory. This transition from discrete layer stacking to continuous evolution enhances the model's physical intuitiveness and interpretability. For example, ODE-based models like ResNets [90] and LM-ResNets [164] perceive inter-layer evolution as a discretization of continuous flows, whereas PDE-based models such as PDE-CNNs [211] encode geometric structures and local features through spatial derivatives. Additionally, DDEs facilitate deterministic generative modeling, including Neural ODEs [29] and flow matching methods [147, 148]. Neural ODEs generate data by solving initial value problems, providing benefits like reversibility and adaptive-depth computation; flow matching methods construct continuous trajectories between distributions by deriving velocity fields through optimal transport. These approaches underscore the pivotal role of DDEs in the unification of neural architectures and generative modeling.

SDEs extend ODEs by incorporating diffusion terms (noise), transforming deterministic systems into stochastic ones. This stochasticity induces trajectory variability from identical initial states, improving generalization by capturing richer distributional variations and enhancing reconstruction, which enables the generation of diverse, high-quality samples. Randomness is pervasive in deep neural networks, manifested through mechanisms like weight initialization [131, 184], minibatch sampling [40], and stochastic regularization methods such as Dropout [9, 65, 132, 213] and Gaussian noise injection [20, 21, 92]. These strategies profoundly influence a model's generalization performance, optimization dynamics, and robustness. Recent investigations strive to rigorously delineate the role of randomness in deep networks via SDEs, giving rise to two primary research trajectories. One trajectory emphasizes the explanation and design of stochastic regularization techniques: embedding randomness explicitly within networks using SDE frameworks (e.g., Dropout [96, 232] and stochastic depth [104]), which offer coherent mathematical models for network evolution under random perturbations [66, 164, 256]. Another trajectory employs SDEs to model or inform the dynamics of neural networks [11, 22, 24, 111, 120, 143, 155, 165, 192, 230], subdivided into two further categories: Neural stochastic differential equations (Neural SDEs) [155], and stochastic flow-based models, exemplified by Denoising diffusion probabilistic models (DDPMs) [230].

In this work, we present a comprehensive framework for systematically organizing and analyzing deep learning models driven by DDEs and SDEs. Firstly, in the aspect of ODE-inspired deep learning, we provide a detailed summary and categorization from two perspectives: neural network architecture design and deterministic dynamics modeling. Secondly, regarding SDE-inspired deep learning, we systematically outline key advances in stochastic regularization

methods for neural networks and stochastic dynamics modeling. Finally, we conduct an in-depth comparative analysis of the performance of DE-based models across multiple experimental tasks, and discuss important challenges and future research directions in this rapidly evolving field.

2 Preliminaries

2.1 Fundamentals of Deep Learning

The forward propagation of a DNN can be mathematically formulated as a sequence of affine transformations followed by nonlinear activations, i.e.,

$$\begin{aligned} \text{Input : } & A^{(0)} = X, \\ \text{Hidden Layers : } & A^{(l)} = f^{(l)} \left(W^{(l)} A^{(l)} + b^{(l)} \right) \quad (1 \leq l \leq L-1), \\ \text{Output : } & \hat{Y} = f^L \left(W^{(L)} A^{(L-1)} + b^{(L)} \right) \quad (l=L), \end{aligned} \tag{1}$$

where X is the input, $W^{(l)}$ and $b^{(l)}$ are the weight matrix and bias of layer l , and $f^{(l)}(\cdot)$ denotes the activation. Hidden layers typically use ReLU for its optimization benefits [1], while outputs adopt task-specific functions such as Sigmoid for binary classification [86] or Softmax for multi-class tasks [170].

In the deep learning, the training process can be formally represented as a loss minimization task, where the primary objective is optimizing network parameters to minimize empirical risk over a given dataset [77]. Mathematically, this optimization problem is typically high-dimensional and non-convex, necessitating gradient-based iterative methods to effectively traverse parameter spaces and minimize prediction errors [77, 134, 135, 235]. Regularization strategies such as weight decay, dropout, and batch normalization are employed to enhance model generalization and mitigate overfitting [107, 130, 232]. Formally, this optimization problem is defined as:

$$\arg \min_{\theta} \mathcal{L}(\theta; X, Y), \quad \text{s.t. } Y = f(X; \theta), \tag{2}$$

where θ denotes network parameters, $\mathcal{L}(\theta; X, Y)$ is the loss function, X and Y represent training data pairs, and f characterizes the neural network function mapping inputs X to outputs under parameters θ .

As formalized in (2), deep learning fundamentally revolves around two core components: the mapping function f and the loss function \mathcal{L} . The design of neural architectures involves creating expressive, high-dimensional nonlinear mappings that capture complex input-output relationships. Architectural expressiveness is influenced by factors such as network depth, connectivity, activation functions, and parameter scales [90, 103, 135]. Concurrently, the loss function serves as the optimization objective that guides network training by quantifying the discrepancy between predictions and ground-truth labels [77]. Minimizing the loss aligns the learned model with underlying data distributions, balancing empirical risk with generalization capacity [15].

The synergistic interaction between architectural innovation and tailored loss functions drives progress in deep learning. Notable architectures, including convolutional neural networks (CNNs) [129] for spatial inputs, recurrent neural networks (RNNs) [98] for sequential data, and transformers for long-range dependencies [247], significantly enhance modeling capabilities. Simultaneously, specialized loss functions such as cross-entropy for classification [169], mean squared error for regression [46], and adversarial losses for generative modeling precisely align models with diverse task requirements [78]. Thus, optimizing the coordination among architectural design, loss formulation, and data quality remains essential for advancing model performance and robustness.

2.2 Stochastic Differential Equations

Ordinary differential Equations serve as a fundamental mathematical framework for modeling the evolution of deterministic dynamical systems governed by a single independent variable. A general form of an ODE can be expressed as:

$$F\left(x, y, \frac{dy}{dx}, \frac{d^2y}{dx^2}, \dots, \frac{d^ny}{dx^n}\right) = 0, \quad (3)$$

where x denotes the independent variable, $y = y(x)$ is the unknown state function and $\frac{d^k y}{dx^k}$ represents the k -th derivative of y with respect to x . The function F defines a deterministic evolution law. Essentially, an ODE characterizes the dynamical behavior of a system through the rate of change of its state variables. The solution to an ODE is uniquely determined by the initial conditions. Based on the order of the equation and the dimensionality of the system, ODEs can be categorized into the following types [16]:

2.2.1 First-Order ODEs. A first-order ODE is an ODE in which the highest derivative involved is the first derivative, commonly represented as:

$$\frac{dy}{dx} = f(x, y), \quad (4)$$

where $y = y(x)$ is a scalar function of a single variable. The solution to a first-order ODE yields a family of integral curves with one integration constant. Depending on $f(x, y)$, such equations may be linear, nonlinear, separable, or homogeneous, and they are widely applied in modeling single-degree-of-freedom systems, including population growth and chemical reaction rates [263].

2.2.2 Higher-Order ODEs. Higher-order ODEs are those ODEs whose order n satisfies $n > 1$. A general n -th order ODE takes the form:

$$F\left(x, y, \frac{dy}{dx}, \dots, \frac{d^ny}{dx^n}\right) = 0 \quad (5)$$

The general solution involves n independent integration constants and requires n initial conditions to determine a unique particular solution. Such equations may be linear or nonlinear and are typically used to describe multi-degree-of-freedom systems.

2.2.3 Systems of Ordinary Differential Equations (ODESs). ODESs involve two or more interconnected ODEs, capturing the joint dynamics of multiple variables within a complex system. The standard form of a first-order linear system is given by:

$$\frac{dy}{dt} = A(t)y + b(t), \quad (6)$$

where $y = [y_1(t), y_2(t), \dots, y_n(t)]^T$, with $A(t)$ the state matrix and $b(t)$ the input vector. Such systems describe interactions among multiple state variables, and any n -th order ODE can be reformulated as a first-order system of dimension n . They may be linear or nonlinear, often involving cross-dependencies among derivatives, as in circuit models governed by Kirchhoff's laws [52] or biological systems such as the Lotka–Volterra equations [183].

2.2.4 Partial Differential Equations. PDEs model the relationships between partial derivatives of multi-variable functions [59]. They are essential for describing physical phenomena like heat diffusion, wave propagation, and fluid dynamics, where system behavior depends on multiple independent variables, such as space and time [95]. A general PDE is expressed as:

$$F\left(x, t, u, \frac{\partial u}{\partial t}, \frac{\partial u}{\partial x}, \frac{\partial^2 u}{\partial t^2}, \dots\right) = 0, \quad (7)$$

where $u = u(x, t)$ is the unknown function, with t as time and x the spatial coordinates. PDEs describe spatiotemporal dynamics of field variables and are well-posed under suitable initial and boundary conditions. They are classified by order, with second-order PDEs further divided into elliptic, parabolic, and hyperbolic types [233]. PDEs form the foundation of physics, engineering, and many scientific domains.

While ODEs and PDEs have been successful in modeling deterministic systems, their deterministic nature is limited when addressing systems with stochastic perturbations. This limitation is significant given the prevalence of random fluctuations in real-world phenomena, such as financial market uncertainties [94], thermal-driven molecular motion in biological systems [56], and environmental noise in ecological models [175].

2.2.5 Stochastic Differential Equations. SDEs generalize ODEs by incorporating stochastic terms to model systems affected by randomness noise, or uncertainty [245]. Essential for modeling real-world phenomena like financial markets, biological processes, and turbulent flows, SDEs overcome limitations of purely deterministic approaches. Built on stochastic calculus using Itô or Stratonovich integrals [6, 193], a general SDE takes the form:

$$dx = f(x, t) dt + g(x, t) dW(t), \quad (8)$$

where $f(x, t)$ governs deterministic dynamics (drift), while $g(x, t)$ scales random fluctuations driven by Wiener noise $dW(t)$. The diffusion term $g(x, t)$ may be additive (state-independent) or multiplicative (state-dependent), with the latter creating noise-system coupling. For systems with jump dynamics [33] or heavy-tailed noise, the Wiener process can be extended to Lévy processes $d\mathcal{L}(t)$.

Rather than producing deterministic trajectories, SDEs describe the evolution of probability densities over time. The evolution of the probability density function $p(x, t)$, which governs the stochastic dynamics of the diffusion process, is rigorously characterized by the **Fokker-Planck Equation (FPE)** [207]:

$$\frac{\partial p(x, t)}{\partial t} = -\nabla_x \cdot (f(x, t) p(x, t)) + \frac{1}{2} \nabla_x^2 [g^2(x, t) p(x, t)], \quad (9)$$

which can capture two critical phases of the diffusion process:

- **Initialization Phase ($t = 0$):** The FPE is initialized with the true data distribution $p(x, 0)$, thereby anchoring the diffusion dynamics to the intrinsic structure of the input data manifold.
- **Asymptotic Convergence ($t \rightarrow \infty$):** As time progresses, the steady-state solution of the FPE converges to a tractable prior distribution, typically a standard Gaussian $p(x, \infty) \sim \mathcal{N}(0, I)$. This convergence ensures the feasibility of reverse-time sampling in generative modeling, as the terminal noise distribution permits efficient sampling and analytical inversion for trajectory reconstruction.

By removing the stochasticity, the evolution of the probability density $p(x, t)$ is governed by the following Liouville equation

$$\frac{\partial p(x, t)}{\partial t} = -\nabla_x \cdot (f(x, t) \cdot p(x, t)), \quad (10)$$

which can be viewed as a special case of the FPE with vanishing diffusion.

The FPE provides a rigorous mathematical framework for describing uncertainty evolution in systems governed by SDEs and underpins recent advances in stochastic flow-based models [206]. Solving SDEs is challenging due to the lack of a universal solution framework, with methods broadly falling into three categories: 1). exact solutions via Itô calculus [193]; 2). analytical approximations [67]; 3). numerical schemes [79]. The latter includes Euler–Maruyama [171],

Milstein's method [110], and stochastic Runge–Kutta formulations [176], each balancing accuracy and stability for specific computational needs.

Its reverse process of SDEs (8) is to move backward in time. According to [5], we can get the reverse SDEs. The reverse SDEs can be written as:

$$dx = \left[f(x, t) - g^2(x, t) \nabla_x \log p(x, t) \right] dt + g(x, t) d\bar{W}(t), \quad (11)$$

where $p(x, t)$ denotes the probability distribution of x at time t , and \bar{W} represents the time-reversed Wiener process. From a modeling perspective, ODEs, PDEs, and SDEs offer three fundamental paradigms for characterizing system

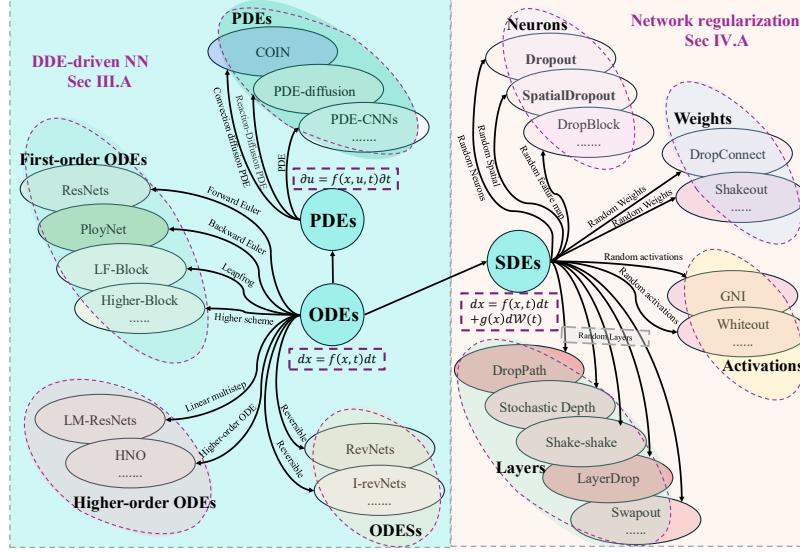


Fig. 1. Overview of DDE-driven neural network architectures (left, Sec.3.1) and stochastic regularization methods inspired by SDEs (right, Sec.4.1).

dynamics. ODEs describe the deterministic evolution of system states along a single temporal trajectory, making them well-suited for modeling sequential or causally structured processes. PDEs extend this framework by incorporating spatial variables, enabling the representation of spatiotemporal dynamics and local interactions, particularly in systems with spatial structure or multiscale phenomena. Building upon this, SDEs introduce stochastic perturbations into the evolution process, capturing intrinsic randomness and external uncertainties commonly observed in real-world systems. Collectively, these approaches represent an evolution from temporal modeling towards comprehensive spatiotemporal modeling, extending from deterministic frameworks to stochastic contexts. They form the foundational principles underpinning continuous dynamical system modeling.

Establishing connections between DEs and deep learning has emerged as a key direction in contemporary research. ODEs and SDEs serve as theoretical foundations for both network design and regularization in discrete models, as illustrated in Figure 1. In the continuous regime, Neural ODEs and flow-based models leverage DEs to model data evolution and enable flexible generative modeling, as shown in Figure 2. This integration enhances the interpretability and robustness of neural networks, and continues to inspire new advances at the intersection of dynamical systems and deep learning.

3 From DDEs to Deep Neural Networks

Since the pioneering work by E et al. [257] that reinterpreted the essence of neural networks from a dynamical systems perspective, DE-guided neural networks design has emerged as a significant research direction [29, 74, 85, 133, 164, 211, 234, 287]. This theoretical breakthrough not only established a rigorous mathematical framework for deep learning but also led to novel architectures that combine robustness and interpretability, substantially enhancing model adaptability in complex real-world applications.

To systematically incorporate the mathematical rigor of ODEs into deep learning, current mainstream approaches can be categorized into two key paradigms:

- **DDE-Driven Architecture:** Exemplified by ResNets [90] and their theoretically refined variant LM-ResNets [164], this paradigm formulates the forward propagation of neural networks as numerical solutions to initial value problems. By employing discretization methods such as Euler or Runge-Kutta schemes [119], it interprets network layers as numerical approximations to DDE solutions. This theoretical perspective not only reveals the function approximation nature of deep networks but also guides the engineering of novel architectures.
- **Deterministic Dynamics Modeling:** Encompassing Neural ODEs [29] and Flow-based models [147], this approach directly describes the continuous evolution of data in latent space through ODE systems. Compared to discrete-time modeling, this continuous dynamics framework demonstrates unique advantages in tasks such as sequential data processing and probability density estimation. Its intrinsic temporal continuity allows for more flexible trajectory interpolation, while the invertibility of ODEs enables more precise gradient computation for backpropagation.

3.1 DDE-Driven Architecture

We first present a structured classification of DDE-driven approaches for deep learning architecture design. A detailed summary of representative works is provided in Table 1, and the corresponding architectural illustrations are presented in Figure 3.

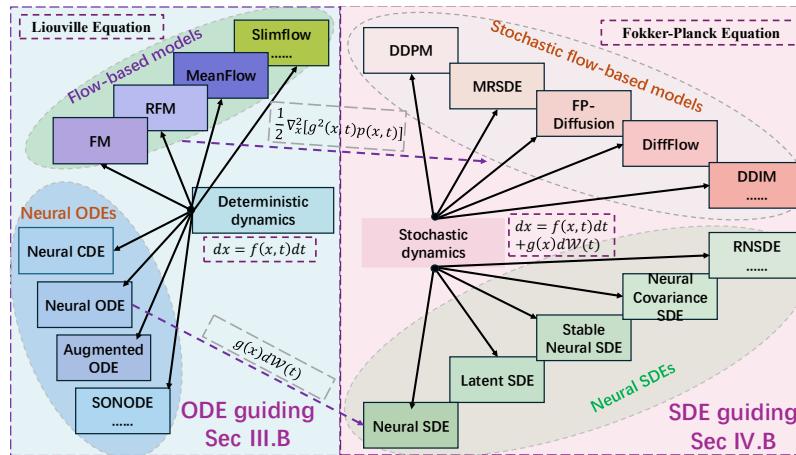


Fig. 2. Overview of ODE-guided deterministic dynamics (left, Sec. 3.2) and SDE-guided stochastic dynamics in deep generative modeling (right, Sec. 4.2).

Table 1. Representative deep learning architectures driven by DDEs, categorized by first-order ODEs, higher-order ODEs, DESs, and PDEs.

Category	Model	Formulation	Year
First-order ODEs	ResNets [90]	$x_{n+1} = x_n + f(x_n)$	2016
	RiR [238]	$x_{n+1} = x_n + f(x_n, t_n), t_{n+1} = g(x_n, t_n)$	2016
	FractalNet [133]	$x_{n+1} = k_1 x_n + k_2 (k_3 x_n + f_1(x_n)) + f_2(k_3 x_n + f_1(x_n))$	2016
	ResNeXt [262]	$x_{n+1} = x_n + \sum_i f_i(x_n)$	2017
	PloyNet [287]	$x_{n+1} = (I - \Delta t f)^{-1} x_n$	2017
	LF-Block [91]	$x_{n+1} = x_{n-1} + 2h f(x_n)$	2019
	Higher-block [166]	$k_1 = f(x_n), k_2 = f(x_n + \frac{1}{2}k_1), k_3 = f(\frac{1}{2}k_2 + x_n), k_4 = f(k_3 + x_n)$ $x_{n+1} = x_n + k_2$ (Midpoint) $x_{n+1} = x_n + \frac{h}{2}(k_1 + f(x_n + k_1))$ (RK2) $x_{n+1} = x_n + \frac{1}{6}(k_1 + k_2 + k_3 + k_4)$ (RK4)	2022
Higher-order ODEs	LM-ResNets [164]	$x_{n+1} = (1 - k)x_n + kx_{n-1} + f(x_n)$	2018
	HNO [126]	$x_{n+1} = (\xi_n)_0 x_n + (\xi_n)_1 x_{n-1} + \dots$	2022
DE Systems	RevNets [74]	$y_{n+1} = y_n + f(x_n), x_{n+1} = x_n + g(y_{n+1})$	2017
	Hamiltonian Network [26]	$y_{n+1} = y_n + f(x_n), x_{n+1} = x_n - g(y_{n+1})$	2018
	I-revNets [12]	$y_{n+1} = y_n + f(x_n), x_{n+1} = P(x_n) + g(y_{n+1})$	2019
PDEs	PDE-CNNs [211]	$F_{\text{sym}}(\theta, x) = -K(\theta)^T \sigma(\mathcal{N}(K(\theta)x, \theta))$ $x_{n+1} = x_n + F_{\text{sym}}(\theta_n, x_n)$ (Parabolic CNN) $x_{n+1} = 2x_n - x_{n-1} + F_{\text{sym}}(\theta_n, x_n)$ (Second-order CNNs)	2020
	LeanConvNets [57]	$x_{n+1} = x_n + f(x_n, \theta_n)$	2020
	PDE-diffusion [113]	$\frac{\partial H}{\partial t} = \nabla \cdot (D \nabla H) + \nabla \cdot (vH) + f(I)$	2022
	PDEs-DNNs [4]	$x_{n+1} = x_n - K^T \phi(Kx_n)$	2023
COIN [256]		$\frac{\partial u(x,t)}{\partial t} = v(x,t) \cdot \nabla u(x,t), t \in [0, T-1]$ $\frac{\partial u(x,t)}{\partial t} = \sigma^2 \Delta u(x,t), t \in [T-1, T]$	2025
		$u(x, 0) = f(x)$	

3.1.1 **ResNets** [90]. ResNets revolutionized deep learning by introducing skip connections to overcome the degradation problem in very DNNs. The fundamental building block implements a residual transformation through the update rule:

$$x_{n+1} = x_n + f(x_n), \quad (12)$$

where x_n denotes the feature representation at layer n , and f is a learnable transformation, typically implemented via convolutional blocks with normalization and nonlinear activations. These skip connections stabilize gradient propagation and improve information flow across layers, enabling the training of very deep networks such as ResNet-50.

Beyond their empirical success, ResNets have been rigorously analyzed from a dynamical systems perspective. Haber et al. [85] and E et al. [257] established that the residual update in (12) can be viewed as a forward Euler discretization of an underlying continuous-time ODE:

$$\frac{dx(t)}{dt} = v(x, t), \quad x(0) = x_0, \quad t \in [0, T], \quad (13)$$

where $v(x, t)$ denotes a velocity field satisfying $v(x, t) = f(x)/\Delta t$. This connection enables the application of numerical analysis tools to study network stability and informs the design of continuous-depth models that leverage ODE solvers for flexible and adaptive computation. Such insights have inspired numerous ResNets variants and advanced our theoretical understanding of deep network architectures.

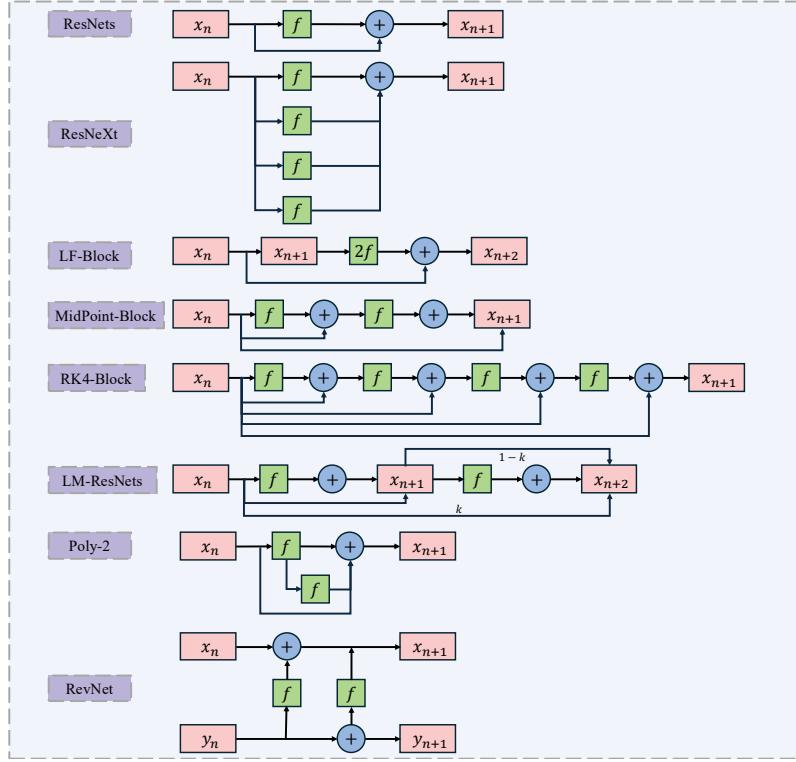


Fig. 3. Schematics of representative network architectures inspired by discretizations of DEs. The diagrams illustrate the computational pathways and connections characteristic of each architecture.

Since the introduction of ResNets, several variants have been proposed to enhance its performance. Architecturally, Wide ResNet (WRN) [278] and ResNeXt [262] increase network width and cardinality, while ResNet-S and ResNet-D [283] focus on optimizing downsampling paths. These adaptations balance depth and width for more efficient training. Numerically, variants like Residual-in-Residual Network (ResInResNet) [238], Polynomial Network (PolyNet) [287], and Fractal Network (FractalNet) [133] explore different formulations to improve stability and expressiveness. Methods such as leapfrog block (LF-Block) [91], Runge-Kutta-2 block (RK2-Block) [91], and higher-order RK4/RK8 [166] integrate time-stepping techniques from numerical analysis to enhance learning accuracy and convergence.

3.1.2 Linear Multistep ResNets [164]. LM-ResNets extend ResNets by introducing multistep dependencies across layers, inspired by Linear Multistep Methods (LMMs) in numerical ODE solvers. Unlike standard ResNets, which update solely from the current state \$x_n\$, LM-ResNets compute \$x_{n+1}\$ as a nonlinear function of \$x_n\$ and past states (e.g., \$x_{n-1}\$), with all dependencies learned by neural modules. The layer update follows the general form of LMMs:

$$x_{n+1} = (1 - k_n)x_n + k_n x_{n-1} + f_n(x_n),$$

where \$k_n \in \mathbb{R}\$ is a trainable parameter. LM-ResNets can be interpreted as a high-order ODE, it can be written as:

$$f(x_n, t) = (1 + k_n)\dot{x}_n + (1 - k_n)\frac{\Delta t}{2}\ddot{x}_n$$

Later on, Kong et al. proposed HNO [126], a high-order numerical architecture that generalizes deep unfolding networks as ODE solvers and enhances them using linear multistep methods for faster convergence and improved accuracy. In parallel, DenseNet [103] introduces a densely connected topology where each layer receives inputs from all preceding layers via feature concatenation, effectively improving feature reuse and gradient flow.

3.1.3 Reversible Residual Networks (RevNets) [74]. Reversible architectures, such as RevNets [74], invertible Residual Networks (i-ResNets) [12], and invertible Reversible Networks (i-RevNets) [109], address the memory bottleneck in deep networks by reconstructing activations during backpropagation, thereby eliminating the need to store intermediate states. RevNets specifically targets memory reduction by making the residual blocks reversible. Unlike traditional ResNets, which require storing intermediate activations, RevNets enables the reconstruction of activations during backpropagation, thus significantly reducing memory usage. This can be formally expressed as:

$$y_{n+1} = y_n + f(x_n), \quad x_{n+1} = x_n + g(y_{n+1}),$$

where f and g are learnable residual functions. The inputs are reconstructed as:

$$x_n = x_{n+1} - g(y_{n+1}), \quad y_n = y_{n+1} - f(x_n).$$

RevNets reduce memory usage with limited performance loss but incur higher computational cost. I-ResNets [12] achieve full-network invertibility for both generative and discriminative tasks, while i-RevNets [109] add nonlinear mappings to further improve memory efficiency. Hamiltonian Networks [26, 80] extend RevNets by modeling reversible dynamical systems, enabling stable, memory-efficient training and accurate physical simulations.

3.1.4 PDE-CNNs [211]. In (12), the function f was treated without structural constraints. For tasks such as speech, image, or video processing, where local spatial transformations are essential, layers typically use convolutions. To formalize this, Ruthotto et al. [211] proposed a PDE-based framework for residual CNNs, modeling forward propagation as nonlinear diffusion. PDE-CNNs discretize this process via the forward Euler method:

$$\frac{\partial x(\theta, t)}{\partial t} = f(\theta, x, t), \quad x(\theta, 0) = x_0, \quad f(\theta, x, t) = K_2(\theta^{(3)}) \sigma\left(N(K_1(\theta^{(1)})Y, \theta^{(2)})\right)$$

where $f(\theta, x, t)$ represents convolution, nonlinearity, and normalization, with specific parameterizations for each operation.

PDE-CNNs [211] are classified into parabolic networks modeling diffusion and hyperbolic networks modeling wave dynamics, including Hamiltonian and second-order variants. PDE-based designs improve stability, interpretability, and efficiency, though deep architectures face higher computational costs. To mitigate this, LeanConvNets [57] and Global Feature Layers [113] reduce depth and complexity, while PDE formulations have also been extended to ResNets, LM-ResNets, and RNNs for greater stability and parameter efficiency [4]. Convection-Diffusion Networks (COIN) [256] further enhance generalization under perturbations, underscoring the robustness of PDE-inspired architectures.

3.2 Modeling deterministic dynamics

We next provide a systematic categorization of deterministic dynamical modeling approaches, focusing on Neural ODEs and Flow-based models and their key variants. Both frameworks can be formulated using (4). Neural ODEs [29] are particularly effective for modeling sequential and continuous-time data, as they learn the governing dynamics $f_\theta(x, t)$ through neural networks parameterization. In contrast, Flow-based models [147] focus on learning the velocity field $v_\theta(x, t)$ in the Liouville equation (10) via neural networks, thereby enabling direct control over the evolution of

probability densities. Representative formulations and recent advances in Flow-based models, along with the details and taxonomy of Neural ODEs, are summarized in Table 3 and Table 2, respectively.

Table 2. Summary of representative neural differential equation models, including NODE-based models, NODE variants, and NSDE-based models.

Category	Model	Formulation	Year
NODE-based models	Neural ODEs [29]	$dx(t) = f_\theta(x, t)dt$	2018
	ANODEs [55]	$da(t) = f_\theta(a, t)dt, a^T(0) = [x_0, 0]$	2019
	SONODEs [190]	$\ddot{x}(t) = f_\theta(x, \dot{x}, t)$	2020
NODE Variants	Neural jump ODE [93]	$h'_i = \text{ODESlove}(f_\theta, h_{i-1}, (t_{i-1, t-t_{i-1}}))$ $h_i = \text{RNNCell}(h'_i, x_i)$	2021
	Neural CDEs [122]	$dz(t) = f_\theta(z) dX(t), X(t)$ is control path	2020
	Neural RDEs [180]	$dz(t) = f_\theta(z(t))d\text{LogSig}_{[r_i, r_{i+1}]}(X)$	2021
NSDE-based models	Neural FDEs [36]	$D^\alpha z(t) = f_\theta(x, t), 0 < \alpha \leq 1$	2025
	Neural SDEs [155]	$dz(t) = f_\theta(z, t)dt + g_\theta(z, t)dW(t)$	2019
	Neural Jump SDEs [111]	$dz(t) = f_\theta(z, t)dt + \omega_\theta(z, k, t) \cdot dN(t)$	2019
NSDE-based models	Latent SDE [143]	$d\tilde{z}(t) = h_\theta(\tilde{z}, t)dt + \sigma(\tilde{z}, t)dW(t)$ (prior) $dz(t) = h_\phi(z, t)dt + \sigma(z, t)dW(t)$ (approximate posterior)	2020
	Neural SDEs as GAN [120]	$dx(t) = \mu_\theta(x, t)dt + \sigma_\theta(x, t)dW(t)$ (generator) $dh(t) = f_\phi(h, t)dt + g_\phi(h, t)dY(t)$ (discriminator)	2021
	Neural Covariance SDE [139]	$d\phi_t(x) = -\frac{1}{2} \int \sum_i (x, x') \nabla \phi_t(x') dx' dt + \sqrt{\sum_i (x, x')} dW(t)$	2022
NSDE-based models	RNSDE [198]	$dx(t) = V(x, t)dt + \sum_{i=1}^m G_i(x, t) dW(t)$	2022
	Stable Neural SDEs [192]	Neural LSDE: $dz(t) = \gamma(\bar{z}; \theta_\gamma)dt + \sigma(\theta_\sigma; t)dW(t)$ Neural LNSDE: $dz(t) = \gamma(\bar{z}, t; \theta_\gamma) + \sigma(\theta_\sigma; t)z dW(t)$ Neural GSDE: $\frac{dz(t)}{z(t)} = \gamma(\bar{z}, t; \theta_\gamma)dt + \sigma(\theta_\sigma; t)dW(t)$	2024

3.2.1 Neural ODEs [29]. Neural ODEs overcome the limitations of discrete architectures by modeling continuous-time dynamics and enabling adaptive computation. Unlike ResNets, which rely on fixed transformations across layers, Neural ODEs describe the evolution of hidden states through an ODE:

$$\frac{dx(t)}{dt} = f_\theta(x, t), \quad (14)$$

where f_θ defines the vector field governing the dynamics. Instead of discrete layers, the hidden state evolves continuously via an ODE solver, with gradients computed using the adjoint method [202], enabling memory-efficient training.

The continuous-time paradigm offers advantages such as constant memory cost regardless of model depth, suitability for irregularly sampled time series [29, 209], and adaptive computational depth. However, Neural ODEs come with drawbacks, including high computational cost during training and inference, as well as challenges with numerical instability and sensitivity to integration dynamics [29, 55, 118, 189].

Augmented Neural ODEs (ANODEs) [55] overcome the expressivity constraints of standard Neural ODEs by modeling system dynamics in an extended state space. Rather than operating solely on the original state variable $x(t)$, ANODEs define the system over an extended state $a(t) = [x(t), z(t)]$, where $z(t)$ is an auxiliary variable. The dynamics are given by:

$$\frac{da(t)}{dt} = f_\theta(a, t), a^T(0) = [x_0, z_0],$$

Table 3. Overview of ODE- and SDE-guided generative models.

Flow-based models	Model	Formulation	Year
Generalized Flow matching (Deterministic)	Flow matching [147]	$dx(t) = v_\theta(x, t) dt$	2022
	CFM [147]	$dx(t) = v_\theta(x, t) dt, x(1) \sim p_{\text{target}}(x y)$	2022
	RFM [28]	$dx(t) = v_\theta^M(x, t) dt, x \in \mathcal{M}$	2023
	DFM [69]	$dp(t) = Q_\theta(t)^\top p(t) dt$	2024
	Fisher-Flow [47]	$dp(t) = v_\theta(p, t) dt, v_\theta(p, t) = g_p^{-1} \nabla_p \mathcal{L}(p, t)$	2024
	SFM [32]	$d\mu = v_\theta(\mu, t) dt, \mu \in \Delta^{n-1}$	2024
	Generator Matching [99]	$dp(t) = \mathcal{G}_\theta^*(p, t) dt, \mathcal{G}_\theta^* \text{ is a generalized generating operator}$	2025
	CS-DFM [31]	$dx(t) = v_\theta(x, t) dt, x(t) = \pi^{(\alpha)}(\mu)$	2025
Trajectory Optimization (Deterministic)	DiffoCFM [37]	$dz(t) = v_\theta^E(z, t, y) dt, dx(t) = v_\theta^M(x, t, y) dt, x(t) = \varphi^{-1}(z)$	2025
	Rectified Flow [152, 154]	$dx(t) = (x(1) - x(0)) dt$	2022
	InstaFlow [157]	$x(1) = x(0) + v(x(0) T)$	2023
	PeRFlow [269]	$dx(t) = (x(1)^{(k)} - x(0)^{(k)}) dt, t \in [t_{k-1}, t_k]$	2024
	Hierarchical Rectified Flow [289]	$dx^{(l)}(t) = (x(1)^{(l)} - x(0)^{(l)}) dt, l = 1, 2, \dots, L$	2025
	MeanFlow [70]	$x(1) = x(0) + \int_0^1 v_\theta(x, t) dt \approx x(0) + \bar{v}(x(0))$	2025
Stochastic Flow-based models	Slimflow [295]	$x(1) = x(0) + v_\theta(x(0))$	2025
	Additive SDE	$dx = f(x, t) dt + g(t) dW(t)$	
	NCSN [228]	$dx = \sqrt{\frac{d(\sigma(t))^2}{dt}} dW(t)$	2019
	DDPMs [97]	$dx = -\frac{1}{2}\beta(t)x dt + \sqrt{\beta(t)} dW(t)$	2020
	MSDiff [11]	$dx = \mu(x, t) dt + \sigma_x(t) dW(t)$ $dy = \mu(y, t) dt + \sigma_y(t) dW(t)$	2021
	MRSDE [165]	$dx = \theta(t)(\mu - x) dt + \sigma(t) dW(t)$	2023
	BFNs [186]	$dx = xf(t) dt + \sigma(t) dW(t)$	2023
	Lévy – Itô Model (LIM) [274]	$dx = f(x, t) dt + g(t) dL(t)$	2023
	OU diffusion [22]	$dx = -x(t) dt + \sqrt{2}dW(t)$	2024
	Multiplicative SDE	$dx = f(x, t) dt + g(x, t) dW(t)$	
Forward SDE (Stochastic)	SDMs [205]	$dY^i = b(\alpha_i - Y^i) dt + \sqrt{2bY^i} dW^i(t), x = Y / \sum_j Y^j$	2022
	FP-Diffusion [54]	$dx = f(x) dt + \sqrt{2R(x)} dW(t)$ $dx = f(x)\beta'(t) dt + \sqrt{2\beta'(t)R(x)} dW(t)$	2023
	General reverse SDE	$dx = [f(x, t) - g(t)^2 \nabla_x \log p(x, t)] dt + g(t) dW(t)$	
	DiffFlow [284]	$dx = [f(x, t) + \beta(t) \nabla \log \frac{g(u(t)x, \sigma(t))}{p(x, t)} + \frac{g^2(t)}{2} \nabla \log p(x, t)] dt$ $+ \sqrt{g^2(t) - \lambda^2(t)} d\bar{W}(t)$	2021
	Unified Framework [24]	$dx = [f(x, t) - \gamma \cdot \frac{1}{2}g^2(t)s_\theta(x, t)] dt + \sqrt{1 - \gamma^2}g(t)d\bar{W}(t)$	2023
	DEIS [285]	$dx = [f(t)x - \frac{1+\lambda^2}{2}g^2(t)s_\theta(x, t)] dt + \lambda g(t)d\bar{W}(t)$	2023
Solvers of Reverse SDE (Stochastic)	ER SDE [41]	$dx = [f(t)x - \frac{g^2(t)+h^2(t)}{2}s_\theta(x, t)] dt + h(t)d\bar{W}(t)$	2025
	PF ODE [230]	$dx = [f(x, t) - \frac{1}{2}g^2(t)s_\theta(x, t)] dt$	2020
	DDIM [227]	$dx = -\frac{1}{2}\beta(t)[x + s_\theta(x)] dt$	2020

which relaxes the topological constraints of the flow and allows the ODEs to learn more complex mappings while preserving uniqueness and continuous-time structure.

Second-Order Neural ODEs (SONODEs) [173, 190] extend Neural ODEs by modeling second-order dynamics. Unlike first-order variants, they incorporate initial velocity and acceleration, yielding more faithful representations of physical systems. Embedding physics-inspired inductive biases further enhances interpretability and numerical stability.

Manuscript submitted to ACM

The general form is given by:

$$\ddot{x} = f_\theta(x, \dot{x}, t), \dot{x}(t_0) = g_\theta(x(t_0)), x(t_0) = X_0,$$

where g_θ and f_θ are neural networks that predict the initial velocity and acceleration, respectively. While the system can be reformulated as a first-order ODE for training purposes, SONODEs demonstrate superior performance over ANODEs in tasks requiring physical interpretability, robustness, and extrapolation.

High computational cost during training and inference remains a key challenge for Neural ODEs, consequently, a range of methods have been proposed to improve their efficiency. [14, 55, 61, 72, 118, 137, 150, 174, 185, 190, 208, 261, 290]. In parallel, to enhance the stability and robustness of Neural ODEs under long-term integration and noisy inputs, several approaches have been developed [35, 43, 48, 115, 146, 155, 172, 241, 268, 279].

Standard Neural ODEs are limited in modeling irregularly sampled time series and in capturing complex temporal dynamics. In response, researchers have developed several important extensions. Among these, Neural Controlled Differential Equations (Neural CDEs) [122] formulate the hidden state evolution as a controlled differential equation, providing:

$$z_t = z_{t_0} + \int_{t_0}^t f_\theta(z_s) dX_s, \quad t \in (t_0, t_n),$$

where X_s is an interpolated input path and f_θ controls how the state responds to input variations. Neural CDEs enable learning from irregularly sampled data by treating inputs as continuous trajectories. Neural Rough Differential Equations (Neural RDEs) [180] extend this methods to rough or non-differentiable paths using rough path theory [63], capturing high-frequency or noisy dynamics through higher-order integration terms. Neural Fractional Differential Equations (Neural FDEs) [36] incorporate long-range memory and non-local dependencies by modeling

$$D^\alpha z(t) = f_\theta(z(t), t), \quad 0 < \alpha < 1,$$

where D^α is a Caputo or Riemann–Liouville fractional derivative. Neural FDEs consider history-dependent dynamics, making them suitable for non-Markovian systems.

3.2.2 Flow-based models [147, 148]. Flow-based models have proven highly effective for generative modeling. The core objective is to deterministically transform samples from a simple base distribution (e.g., Gaussian) to a complex target distribution under the guidance of ODEs. The evolution of probability densities in this setting is governed by the Liouville equation (10), with the neural networks parameterizing the velocity field $v_\theta(x, t)$:

$$\frac{\partial p(x, t)}{\partial t} = -\nabla_x \cdot (v_\theta(x, t) \cdot p(x, t)) \quad (15)$$

Liouville equation (10) describes the evolution of probability density under a deterministic flow. The velocity field $v_\theta(x, t)$ can be defined by ODEs:

$$\frac{dx}{dt} = v_\theta(x, t), \quad x(0) \sim p(0), \quad x(1) \sim p_{\text{target}} \quad (16)$$

Specifically, flow matching defines an explicit guiding velocity field and trains a NN to approximate it, sidestepping explicit density computation via conditional path interpolation and enabling scalable, tractable training.

Despite its effectiveness, the standard formulation of flow matching remains limited in scope, being primarily defined for continuous Euclidean spaces [28] and linear interpolation trajectories [3, 154]. To address these constraints, subsequent research has focused on two major directions: generalizing flow matching to discrete domains [19, 69], graph-structured data [101], and manifold-valued spaces [28, 116]; and developing more efficient transport paths [152, 154], such as direct one-step generation [70], to further enhance sampling efficiency and generative performance.

a) Generalized Flow Matching: A series of generalized flow matching approaches have emerged, each systematically extending the framework toward broader classes of data and structural priors. Notable examples include:

Conditional Flow Matching (CFM) [147] is a method that learns a velocity field $v_\theta(x, t, y)$, which is conditioned on an auxiliary variable y , allowing for controllable generation. The system is governed by the DEs:

$$\frac{dx}{dt} = v_\theta(x, t), \quad x(0) \sim p(0), \quad x(1) \sim p_{\text{target}}(x|y),$$

which facilitates the direct generation of samples with specified attributes to ensure alignment with conditioning information. This capability is particularly beneficial for tasks such as class-conditional image synthesis.

Discrete Flow Matching (DFM) [19, 69] generalizes flow matching to discrete and categorical data domains by replacing the continuous flow with a continuous-time Markov jump process. The dynamics are governed by a time-dependent rate matrix $Q_\theta(t)$:

$$\frac{dp(t)}{dt} = Q_\theta(t)^\top p(t),$$

where $p(t)$ denotes the time-evolving probability mass over discrete states. This approach enables non-autoregressive and efficient generation in sequence, graph, and molecular data.

These developments mark a shift in flow matching research toward modeling data in non-Euclidean [28, 116] and structured spaces [31, 32, 47]. Riemannian Flow Matching (RFM) [28] adapts flow matching to non-Euclidean domains by respecting manifold constraints, enabling generative modeling on curved data spaces. Generator Matching [99] generalizes flow matching to arbitrary modalities and Continuous Time Markov Processes (CTMPs), supporting non-autoregressive generation for discrete sequences and graphs. Methods like Continuous-State Discrete Flow Matching (CS-DFM) [31], Fisher-Flow [47], and Statistical Flow Matching (SFM) [32] bring in information geometry, aligning probability transport with natural statistical metrics such as the Fisher–Rao distance. Diffeomorphism Conditional Flow Matching (DiffeoCFM) [37] further extends flow matching to positive-definite matrix manifolds via pullback geometry. Altogether, these innovations substantially broaden the applicability and expressivity of flow matching, providing a robust geometric foundation for flow-based generative modeling in complex domains.

b) Trajectory Optimization: Trajectory Optimization explores advanced strategies to optimize both the generative trajectory and its numerical integration, aiming to improve sampling efficiency, stability, and accuracy [45, 253, 271].

Rectified Flow [152, 154] simplifies the generative process by constraining the transport path between noise and data to a straight line in data space, allowing for highly efficient and deterministic sampling. The underlying ODE is

$$\frac{dx}{dt} = x(1) - x(0),$$

where $x(0)$ and $x(1)$ are the endpoints of the trajectory, allowing the model to learn a velocity field that maps base samples directly along a rectified path. Recent works have extended Rectified Flow by addressing its training stability and efficiency [82, 136, 167, 197, 216], enabling hierarchical and piecewise trajectories [269, 288], and exploring more flexible path designs beyond strict linearity [252]. These advances further enhance the speed, adaptability, and application scope of the rectified flow framework [141, 295, 296], establishing it as a versatile tool for both efficient and controllable generative modeling.

MeanFlow [70] adopts a one-step generation approach by learning a time-averaged velocity field $\bar{v}_\theta(x, t)$. This enables the model to directly map a base sample to the target in a single step:

$$x(1) = x(0) + \int_0^1 v_\theta(x, t) dt \approx x(0) + \bar{v}(x(0)).$$

Recent models such as InstaFlow [157], Optimal Flow Matching [128], Motion Prediction Conditional Flow Matching (MoFlow) [64], Shortcut Models [62], Distilled Decoding [149] and SlimFlow [295] learn the optimal transport map in a single step. These methodologies enable highly efficient, end-to-end generation by eliminating the need for iterative refinement and extend this approach to a broader range of applications and domains [140].

4 From SDEs to Deep Neural Networks

Introducing stochasticity into ODEs give rise to SDEs 8, which provide a more expressive and realistic framework for modeling uncertainty. SDEs appear in two common forms: additive noise, which accounts for external disturbances such as environmental fluctuations, and multiplicative noise, which captures state-dependent randomness arising from factors like thermal or reaction-level variations.

The deep integration of SDEs and neural networks have opened up a new paradigm in modern machine learning, enabling both more powerful regularization techniques and the principled modeling of stochastic dynamics. This intersection manifests primarily in two domains:

- **Stochastic Network Regularization:** Controlled noise is injected at various levels of the neural networks through the introduction of stochastic regularization mechanisms. This includes techniques such as Dropout and its variants at the neuron level [9, 232], parameter perturbations in the weight space [114, 144], noise injection in the activation function domain [92, 142, 178], and dynamic adjustment of network depth [68, 104]. These methods enhance network robustness and generalization by mimicking stochastic perturbations in the optimization landscape, helping to avoid poor local minima.
- **Stochastic Dynamics Modeling:** The forward propagation in neural networks can be mathematically characterized as a dynamical system driven by SDEs (8). Neural SDEs [155] parameterize both the drift and diffusion coefficients using neural networks, establishing a flexible and unified framework for modeling complex stochastic dynamics in data. In contrast, DDPMs [97, 226] construct a fixed forward diffusion process (by incrementally adding Gaussian noise) and achieve data generation by learning the drift term of the corresponding reverse SDE (11). These two approaches differ fundamentally in their learning objectives: Neural SDEs model the forward-time stochastic dynamics, while DDPMs focus on learning the reverse-time denoising dynamics to generate data from noise.

4.1 Stochastic Network regularization

We begin by reviewing stochastic regularization methods that introduce randomness into different components of neural networks, such as neurons, weights, activations, and architectural paths. Representative methods are summarized in Table 4, along with their key design choices and regularization effects.

4.1.1 Dropout [96, 232]. Dropout is a fundamental regularization technique that randomly deactivates hidden units during training to mitigate overfitting. When integrated with ResNets, the forward propagation through a dropout-augmented residual block is formally expressed as:

$$x_{n+1} = x_n + f(x_n, \omega_n) \odot \frac{z_n}{p}, \quad z_n \sim \text{Bernoulli}(p),$$

where p is the retention probability, \odot denotes element-wise multiplication. Dropout has been theoretically interpreted as an approximation to Bayesian inference [7], with further formalization accomplished through a SDE [156, 256], given

Table 4. Summary of SDE-guided stochastic regularization methods for deep neural network models, categorized by their application to neurons, weights, activations, and layers.

Category	Model	Formulation	Year
Neurons	Dropout [96, 232]	$x_{n+1} = x_n + f(x_n, \omega_n) \odot \frac{z_n}{p}, z_n \sim \text{Bernoulli}(p)$	2012
	Fast Dropout [255]	$x_{n+1} = (1-p)x_n + \mathcal{N}(0, p(1-p)x_n^2)$	2013
	Annealed Dropout [204]	$x_{n+1} = x_n + f(x_n, \omega_n) \odot \frac{z_n(t)}{p}, z_n \sim \text{Bernoulli}(p(t)) \text{ and } p(t) = p(t-1) + \alpha_t(\theta)$	2014
	SpatialDropout [239]	$\tilde{y}_c^{l-1} = z_c \cdot y_c^{l-1}, y_c^{l-1} \text{ is the } c\text{-th channel feature map}$	2014
	Curriculum Dropout [179]	$x_{n+1} = x_n + f(x_n, \omega_n) \odot \frac{z_n(t)}{p}, z_n \sim \text{Bernoulli}(p(t)) \text{ and } p(t) = (1-\theta)(1-e^{-\gamma t})$	2017
	Targeted Dropout [75]	$x_{n+1} = x_n + f(x_n, \omega_n) \odot \frac{z_n}{p}, z_n \sim \text{Bernoulli}(p_n) \text{ and } p_n = \text{TargetDropout}(s_n)$	2018
Weights	DropBlock [71]	$\tilde{A} = A \odot M, M_{i,j} = \text{Bernoulli}(\gamma) \text{ and } A \text{ is the feature map}$	2018
	DropConnect [251]	$\tilde{W} = W \odot M, M_{i,j} = \text{Bernoulli}(1-p)$	2013
	Variational Dropout [125]	$W \sim \mathcal{N}(\mu, \alpha\mu^2), \alpha \text{ is a learnable parameter.}$	2015
Activations	Shakeout [114]	$\tilde{W}_{ij} = c(r_j - 1)S_{ij} - r_j \cdot (\frac{1}{p}W_{ij} + c\frac{1}{1-p}S_{ij}), S_{ij} = \text{sgn}(W_{ij}), r_j \sim \text{Bernoulli}(p)$	2016
	Whiteout [142]	$x_{n+1} = x_n + f(x_n) + \epsilon, \epsilon \sim \mathcal{N}(0, \sigma^2 x ^{2\gamma})$	2016
	GNI [178]	$x_{n+1} = x_n + f(x_n) + \sigma\mathcal{N}(0, I)$	2020
	PNI [92]	$x_{n+1} = x_n + \alpha \cdot \mathcal{N}(0, I), \alpha \text{ is a learnable parameter.}$	2019
Layers	Stochastic Depth [104]	$x_{n+1} = x_n + z_n f(x_n), z_n \sim \text{Bernoulli}(p)$	2016
	DropPath [133]	$x_{n+1} = x_n + \sum_i z_{in} f_i(x_n), z_{in} \sim \text{Bernoulli}(p)$	2016
	Swapout [222]	$x_{n+1} = \alpha x_n + \beta f(x), \alpha, \beta \sim \text{Bernoulli}(p)$	2016
	Shake-shake regularization [68]	$x_{n+1} = x_n + \alpha_n f_1(x) + (1-\alpha_n) f_2(x), \alpha_n \sim \text{Bernoulli}(p)$	2017
	PyramidSepDrop [267]	$x_{n+1} = x_n + \sum_i \alpha_i f_i(x), f(x) = [f_1(x), \dots, f_K(x)], \alpha_i \sim \text{Bernoulli}(1-p)$	2017
	ShakeDrop [266]	$x_{n+1} = x_n + \alpha_n \cdot b \cdot f(x), \alpha_n \sim \text{Bernoulli}(p), b \sim \text{U}([-1, 1])$	2018
	Scheduled drophead [292]	$\text{head}_i^{\text{dropped}} = \alpha_i(t) \cdot \text{head}_i, \alpha_i(t) \sim \text{Bernoulli}(1-p), p(t) = p_{\max} \cdot \frac{t}{T}$	2018
	LayerDrop [60]	$h_{l+1} = \alpha_l \cdot \text{Layer}_l(h_l) + (1-\alpha_l) \cdot h_l, \alpha_l \sim \text{Bernoulli}(1-p)$	2019

by:

$$dx(t) = f(x)dt + \sqrt{\frac{1-p}{p}} f(x) \cdot dW(t), \quad (17)$$

where $W(t)$ has the same meaning as in (8). This perspective enables stability analysis and supports adaptive optimization.

Since the introduction of standard dropout, numerous neuron-level stochastic regularization variants have emerged. Fast Dropout [255] leverages Gaussian approximation for efficiency, while Annealed [204] and Curriculum Dropout [179] adjust noise levels over training. Variational Dropout [125] treats weights as Gaussian variables for adaptive regularization and sparsity. Other extensions include structure-aware methods such as SpatialDropout [239], Max-pooling and Convolutional Dropout [259], as well as selective masking strategies like Targeted Dropout [75], DropBlock [71], and EDropout [214].

4.1.2 DropConnect [251]. DropConnect regularizes neural networks by randomly masking weights rather than neurons. It applies a binary mask M , sampled as $M_{i,j} \sim \text{Bernoulli}(1-p)$, to the weight matrix W , producing an output:

$$y = f_a((W \odot M)x), x_{n+1}^i = \sum_j (W_{ij} x_n^j) M_{ij}$$

where \odot denotes element-wise multiplication and f_a is a non-linear activation function.

Theoretically, DropConnect reduces to Dropout in shallow networks [196] and can be represented by the following SDE:

$$dx(t) = pWx dt + \sqrt{p(1-p)} (W \odot W)^{1/2} (x \odot x)^{1/2} d\mathcal{W}(t),$$

which has inspired numerous extensions, including Sparsity-aware masking [144], Output uncertainty modeling via Dirichlet distributions [291], Gradient-based dynamic drop rates [272], Adaptive drop rates using generalization gap and complexity measures [112].

4.1.3 Shakeout [114]. Shakeout is a regularization method that unifies the strengths of Dropout and L1 regularization (Lasso). By injecting random weight perturbations and encouraging sparsity, it enhances model generalization. The forward pass is defined as:

$$\tilde{W}_{ij} = c(r_j - 1)S_{ij} + r_j \cdot \left(\frac{1}{p}W_{ij} + c \frac{1}{1-p}S_{ij} \right), \quad S_{ij} = \text{sgn}(W_{ij}), \quad r_j \sim \text{Bernoulli}(p),$$

where i and j index the output units of layer $n+1$ and n , respectively, W_{ij} denotes the weight, and c modulates the L1 perturbation intensity when a unit is dropped.

4.1.4 Gaussian Noise Injection (GNI) [178]. GNI enhances model generalization by perturbing intermediate representations during training. In a standard ResNet with L residual blocks, the output of the n -th block with injected noise is given by:

$$x_{n+1} = x_n + f(x_n) + \alpha \mathcal{N}(0, I),$$

where α controls the noise intensity and $\mathcal{N}(0, I)$ is standard Gaussian noise. By setting $\alpha = \sigma\sqrt{\Delta t}$ and interpreting $f(x_n)$ as a discretized velocity field $v(x_n, t_n)$, this update rule corresponds to the Euler–Maruyama discretization of SDEs:

$$dx(t) = v(x, t) dt + \sigma d\mathcal{W}(t).$$

Li et al.[142] proposed Whiteout, which injects adaptive Gaussian noise into activations to promote structured sparsity, later given a variational interpretation with multi-sample estimation[188]. Subsequent analyses revealed that activation-level noise penalizes high-frequency components [21], relates to ridge regularization in the infinite-sample limit [51], and extends to overparameterized regimes [194]. From a robustness perspective, Parametric Noise Injection (PNI)[92] introduced learnable noise at activations or weights, improving adversarial resilience. Overall, Gaussian noise injection (GNI) serves as a principled regularization mechanism, smoothing the loss landscape and connecting discrete training with continuous-time stochastic dynamics[194, 273].

4.1.5 Stochastic Depth [104]. Stochastic Depth is an effective regularization method for deep residual networks that randomly bypasses residual blocks during training. At each iteration, only a subset of layers is active, reducing overfitting and enabling efficient training. For a ResNet with L layers, the n -th block is formulated as:

$$x_{n+1} = x_n + z_n f(x_n), \quad z_n \sim \text{Bernoulli}(p),$$

where p being the layer-wise survival probability. The network with stochastic drop can be written as an SDE

$$dx(t) = pf(x)dt + \sqrt{p(1-p)}F(x)d\mathcal{W}(t)$$

Recent studies have explored the theoretical and empirical properties of stochastic depth [88]. Yamada et al. [267] proposed PyramidSepDrop, which independently applies residual dropping to new and existing channels, enhancing regularization and improving generalization in image classification.

4.1.6 Shake-Shake regularization [68]. Shake-Shake regularization enhances multi-branch neural architectures through stochastic convex combinations of parallel residual branches during both forward and backward propagation. Applied to multi-branch architectures such as ResNets, it provides stronger regularization than standard dropout and improves robustness to perturbations. For a residual block with two parallel branches $f_1(x)$ and $f_2(x)$, the Shake-Shake forward pass is defined as:

$$x_{n+1} = x_n + \alpha_n f_1(x_n) + (1 - \alpha_n) f_2(x_n), \quad \alpha_n \sim \mathcal{U}(0, 1).$$

In the infinitesimal limit, Shake-Shake regularization can be approximated by an SDE, assuming small output perturbations at each block:

$$dx(t) = \frac{1}{2} (f_1(x) + f_2(x)) dt + \frac{1}{\sqrt{12}} (f_1(x) - f_2(x)) dW(t).$$

Building on the idea of structured stochasticity, several extensions have been proposed to regularize deep networks through path-level perturbations. DropPath [133] randomly drops entire residual paths, simplifying the training graph. Swapout [222] applies independent stochastic perturbations to both identity and residual branches, enabling more flexible noise injection. Scheduled DropPath [292] gradually increases the drop probability over training, balancing early stability and later regularization, while LayerPath [60] dynamically adjusts drop strength by network depth for structure-aware regularization. ShakeDrop [266] extends this idea to single-branch residual networks by stochastically blending residual and identity paths, introducing smooth perturbations that further enhance training stability and generalization. Collectively, these methods underscore the effectiveness of structured noise at the path level and provide a unified perspective on regularization and optimization in deep architectures.

4.2 Stochastic Dynamics Modeling

SDEs (8) provide a powerful mathematical framework for modeling random dynamical systems in deep learning. Two major classes of SDE-driven approaches have emerged. The first, Neural SDEs [155] utilize neural networks to parameterize both drift and diffusion terms, providing a flexible framework for modeling complex stochastic dynamics and effectively capturing temporal uncertainty in time-series analysis. A detailed categorization and summary of these methods can be found in Table 2. The second class consists of stochastic flow-based models, represented DDPMs [230]. The evolution of probability densities is described by the FPE (9), and the sampling process can be expressed using the SDE in (8). DDPMs learn the score or noise component to guide the evolution of probability densities, avoiding explicit density computation. Detailed comparisons are summarized in Table 3.

4.2.1 Neural SDEs [155]. Neural SDEs extend Neural ODEs [29] by introducing stochasticity through a learnable diffusion term, enabling the modeling of systems with intrinsic randomness. Unlike the deterministic trajectories of Neural ODEs, Neural SDEs capture uncertainty via stochastic processes parameterized by neural networks. The latent state x_t evolves according to the Itô SDE

$$dx(t) = f_\theta(x, t) dt + g_\theta(x, t) dW(t), \quad (18)$$

where f_θ and g_θ are drift and diffusion networks, respectively. The parameterized SDE (18) can be simulated using numerical methods such as Euler–Maruyama [171]. Training involves handling gradient propagation through stochastic

paths, typically addressed by reparameterization or adjoint methods [202]. Neural SDEs are well-suited for modeling noisy or irregular data, with applications in time series analysis, generative modeling, and financial systems. However, their use often entails high training complexity, significant computational overhead, and limited interpretability.

Recent advancements in Neural SDEs can be categorized into four main areas. Firstly, they have been developed as expressive generative models: Tzen et al. [242] link them to diffusion-limit approximations of latent Gaussian models, Li et al. [143] extend latent ODEs by integrating stochastic priors, and Kidger et al. [120] adapt both the generator and discriminator of GANs as Neural SDEs. Secondly, in terms of efficiency and stability, efforts have concentrated on enhancing solvers and objectives. Reversible solvers [121] and deterministic approximations [159] have been employed to mitigate integration overhead, while alternative objectives such as signature kernel losses [108] and marginal-matching criteria [280] bolster training robustness. Complementary improvements include cubature methods on Wiener space [225], finite-dimensional distribution matching [281], stochastic optimal control formulations [44], and high-dimensional gradient estimators [254], all contributing to enhanced scalability and optimization efficiency. Thirdly, Neural SDEs have been applied across various fields such as robust financial modeling [73], spatiotemporal dynamics [215], and unified sequence modeling [220]. Finally, recent studies have improved the modeling of rare and irregular dynamics: Neural Jump SDEs [111] address discontinuities through Poisson jumps, while Stable SDEs [192] introduce variants like Langevin-type, linear-noise, and geometric models to enhance the analysis of irregular time series. Collectively, these developments position Neural SDEs as a versatile and robust framework for modeling complex stochastic systems in continuous time. A detailed overview is presented in Table 2 for Neural SDEs and their variants.

4.2.2 Stochastic flow-based models [97, 231]. Stochastic flow-based models generate samples by first adding noise to data in a forward process, then reconstructing data from noise in a reverse denoising process. The forward process can be formulated as a SDE:

$$dx = f(x, t) dt + g(t) dW(t), \quad (19)$$

where $f(x, t) = -\frac{1}{2}\beta(t)x$ and $g(t) = \sqrt{\beta(t)}$ in the standard DDPMs setting.

In DDPMs, the generative process is realized by solving the reverse SDE, which simulates the dynamic evolution from the noise distribution to the target data distribution. The evolution of the system's probability density is described by the Fokker-Planck equation (FPE) (9), which accurately characterizes the statistical properties of the underlying stochastic dynamics. The corresponding reverse SDE can be expressed as:

$$dx = \left[f(x, t) - g^2(t) \nabla_x \log p(x, t) \right] dt + g(t) d\bar{W}(t), \quad (20)$$

where the score function $\nabla_x \log p(x, t)$ is typically approximated by a parameterized neural network $s_\theta(x, t)$ and trained using score matching methods. Therefore, the generative process of DDPMs essentially involves the discrete numerical solution of the aforementioned reverse SDE.

Guided by the DDPMs paradigm, the development of stochastic flow-based models can be systematically divided into three major directions: the design of forward SDEs, the design of reverse SDEs, and the development of solvers for reverse SDEs.

a) Forward SDE: In diffusion models, the forward process outlines the transformation of the data distribution into a simple noise distribution. This process is typically represented by a SDE, as shown in Equation (19). By flexibly selecting specific forms for the drift term $f(x, t)$ and the diffusion term $g(t)$, or by extending the diffusion term to a state-dependent function $g(x, t)$ to include multiplicative noise, various diffusion model variants can be developed. Several representative works will be introduced in the following section.

Mean-Reverting SDE (MRSDE) [165] extends the diffusion framework to model processes with a natural tendency to revert to a long-term mean, making it suitable for mean-reverting scenarios. The process is described by:

$$dx(t) = \theta(t)(\mu - x)dt + \sigma(t)dW(t),$$

which can help to capture baseline-reverting trajectories, improves realism for mean-reverting data, and stabilizes long-term generative dynamics. In addition to classical additive noise SDEs, a variety of forward processes have been explored for diffusion models. Early examples such as Noise Conditional Score Networks (NCSNs) [228], Multi-Speed Diffusion (MSDiff) [11], Bayesian Flow Networks (BFNs) [264] and Ornstein-Uhlenbeck diffusion [22] all inject Gaussian noise into the data, each differing in their noise schedules and parameterizations.

When the diffusion term $g(x, t)$ depends on x , the additive SDE framework extends to multiplicative noise SDEs, enabling the modeling of state-dependent stochasticity. This extension addresses more complex phenomena such as variable uncertainty or structural constraints in data.

Simplex Diffusion Models (SDMs) [205] model data evolution on the probability simplex Δ^{d-1} using SDEs. To this end, a diffusion process is first defined in the non-negative orthant \mathbb{R}_+^d :

$$dY_t^i = b(\alpha_i - Y_t^i) dt + \sqrt{2bY_t^i} dW_t^i, \quad i = 1, \dots, d, \quad X(t) = \frac{Y_t}{\sum_{j=1}^d Y_t^j},$$

which naturally preserves non-negativity and normalization constraints to make it well-suited for modeling categorical or compositional data.

Flexible Diffusion Processes (FP-Diffusion) [54] enhances model expressiveness by simultaneously learning the drift $f(x)$ and data-dependent diffusion $R(x)$ through a multiplicative noise SDE, surpassing traditional additive noise methods:

$$dx = f(x)\beta'(t)dt + \sqrt{2\beta'(t)R(x)}dW(t),$$

which allows the model to adapt flexibly to heterogeneous data distributions and to capture more intricate generative dynamics by learning both the drift and diffusion structures.

Recent works have generalized the framework by considering non-Gaussian noise [105, 218, 246, 249, 274], which broadens the class of data distributions that can be modeled. Further extensions introduce physical constraints to the diffusion process, for example, by enforcing boundary conditions or reflecting barriers [160, 163, 186, 276, 277].

b) Reverse SDE: While the reverse SDE (20) provides a principled way to model the generative trajectory from noise to data, its form is inherently tied to the forward process and often constrained in flexibility. To overcome these limitations, recent works have explored mathematically redesigning the reverse SDE, either by learning more expressive drift functions or by jointly adapting the drift and diffusion terms.

Unified Framework [24] introduces a tunable parameter $\gamma \in [0, 1]$ into the reverse SDE, allowing smooth interpolation between deterministic ODEs and stochastic SDEs:

$$dx = [f(x, t) - \gamma \cdot \frac{1}{2}g^2(t)s_\theta(x, t)]dt + \sqrt{1 - \gamma^2}g(t)d\bar{W}(t),$$

which can flexibly balance between sample diversity (stochasticity) and sample fidelity (determinism), and unifies a wide range of generative modeling dynamics within a single formulation.

Diffusion Exponential Integrator Sampler (DEIS) [285] enhances sampling efficiency by decoupling noise scaling from the score-driven drift in the reverse SDE:

$$dx = \left[f(t)x - \frac{1 + \lambda^2}{2} g^2(t) s_\theta(x, t) \right] dt + \lambda g(t) d\bar{W}(t),$$

where the parameter λ allows for adaptive control of noise magnitude, enabling larger integration steps in low-noise regions and thus accelerating the generative process without compromising stability.

Other recent advances, such as DiffFlow [284] and ER SDE [41], further explore hybrid drift formulations and independent tuning of drift and diffusion terms, respectively. These extensions provide additional flexibility for balancing sample quality, diversity, and robustness in high-variance regions.

c) Solvers for reverse SDE: Beyond redesigning the reverse dynamics, an important perspective is to view the generative process as a numerical integration problem. Since sampling in generative problems involve integrating the learned SDEs or ODEs from noise to data, advances in solver design can directly improve both sampling speed and diversity. These methods do not alter the model architecture, but instead accelerate generation and improve precision by developing more efficient numerical solvers.

A variety of solvers have been proposed to approximate the solution of the reverse SDEs or its deterministic ODEs counterpart. For example, the reverse process in Probability Flow ODE (PF ODE) [230] can be formulated as the following ODE:

$$dx = [f(x, t) - \frac{1}{2} g^2(t) s_\theta(x, t)] dt. \quad (21)$$

Several representative solvers have been developed to improve sampling efficiency and quality in practice. Denoising Diffusion Implicit Models (DDIMs) [227] can be interpreted as a deterministic numerical solver for the underlying probability flow ODE, accelerating sampling. Generalized denoising diffusion implicit models (gDDIM) [286] interpolates between deterministic and stochastic sampling. Diffusion Probabilistic Model Solver (DPM-Solver) [161, 162], Pseudo Numerical methods for Diffusion models (PNDMs) [151], Approximate Mean-Direction Solver (AMED-Solver) [293], Stochastic Runge-Kutta Methods (SRK) [260] and Stochastic Adaptive Solver (SA-Solver) [265] apply high-order integration methods to improve sampling efficiency and quality. These solvers enable fast and diverse sample generation with fewer steps.

Furthermore, it is crucial to highlight that DDPM, as a powerful generative model, can also serve as an effective regularization technique. Integrating the diffusion probabilistic model into the training process enables the stochastic denoising mechanism to improve the model's performance, training stability, and generalization ability, all without the need for iterative sampling during inference. This advantage has been demonstrated in various tasks, such as image inpainting [145] and image segmentation [83].

5 Applications and Performance

This section reviews the experimental performance of a range of DE-driven neural networks on CIFAR-10 and CIFAR-100 benchmarks. While Neural ODEs and Neural SDEs are particularly excel in modeling sequential and time-series data, we refer the reader to [187, 191] for detailed results in that domain. Here, our focus is on architectures and generative models informed by DDEs and SDEs. For clarity, we organize the discussion into two categories: This section focuses on two main directions: 1) Neural Network architectures guided by DDEs; 2) Dynamics-based generative modeling driven by ODEs and SDEs, including both flow-based models and stochastic flow-based models. Through a comparison

of recent results, we highlight how these equations-guided approaches have advanced the design and effectiveness of deep learning models for vision tasks.

5.1 DEs based Neural Network Models

We provide the experimental results of networks guided by first-order ODEs, higher-order ODEs, as well as systems of equations and PDEs in Table 5. Different numerical integration schemes correspond to different network architectures. Consistent with convergence order theory in numerical analysis, networks employing higher-order or more sophisticated schemes generally achieve superior performance.

The experimental results highlight a clear progression, where innovations in DEs modeling translate into advances in both network architectures and empirical performance. ResNets exemplifies the explicit Euler discretization of first-order ODEs, where the skip connection directly mirrors the explicit integration step, achieving a strong baseline accuracy of 93.63% on CIFAR-10 and 75.46% CIFAR-100. FitResNet builds on this by introducing adaptive step sizes, analogous to variable-step ODE solvers, and improves accuracy to 94.16% on CIFAR-10 and 76.63% on CIFAR-100, demonstrating increased stability and precision. With further advances in mathematical formulation, FractalNet employs a multi-path architecture inspired by Runge-Kutta integration, while RKCNN adopts the fourth-order Runge-Kutta method. These higher-order discretizations enable more accurate modeling of feature dynamics, resulting in notable improvements, such as RKCNN’s 95.52% on CIFAR-10 and 79.31% on CIFAR-100. PDE-guided models further extend this paradigm by embedding principles from parabolic and hyperbolic equations into the network backbone. By explicitly capturing spatial-temporal dependencies and incorporating physical priors, these networks achieve the highest reported performance, with Parabolic and Hyperbolic PDE models both surpassing 95% on CIFAR-10 and achieving 80.55% and 80.34% respectively on CIFAR-100. In addition to superior accuracy, PDE-guided networks provide enhanced interpretability, stability, and a stronger foundation for generalizing to tasks involving structured or physically-constrained data.

Overall, these DE-driven architectures demonstrate that leveraging numerical integration schemes, higher-order dynamics, and manifold-based formulations can significantly enhance network expressiveness, stability, and accuracy, while providing principled insights into deep model design.

5.2 DEs guided Dynamics Modeling

Table 6 systematically compares representative generative models based on their dynamical framework, covering both ODE-guided flow-based models and SDE-guided stochastic flow-based models. Over time, advancements in mathematical modeling and solver design have led to consistent improvements in both generative quality and sampling efficiency on CIFAR-10, as reflected by the trends in Fréchet Inception Distance (FID) and Number of Function Evaluations (NFE) metrics.

A detailed examination reveals clear differences in methodology and performance. ODE-guided flow-based models emphasize deterministic mappings and trajectory optimization, pushing the boundaries of sampling efficiency. Recent methods such as Generator Matching, Rectified Flow, and MeanFlow have reduced the number of sampling steps to just one while maintaining low FID scores. For example, MeanFlow achieves an FID of 2.92, and Generator Matching further lowers it to 2.36. These approaches effectively overcome the inefficiencies of traditional multi-step sampling by directly learning optimal trajectories or velocity fields. In contrast, SDE-guided stochastic flow-based models leverage stochastic dynamics and advanced reverse process solvers to enhance sample quality and efficiency. Early models such as DDPMs required 1000 sampling steps to reach an FID of 3.21. With the introduction of improved solvers and architectures,

Table 5. Performance comparison of representative DDE-driven neural network architectures on CIFAR-10 and CIFAR-100, categorized by first-order, higher-order, DE-guided, and PDE-guided models.

Category	Model	Publication	Code	Params	Dataset	Test acc. (%)
First-order guided	ResNets [90]	CVPR 2016	[code]	1.14M 1.7M	CIFAR-10 CIFAR-100	93.63 72.24
	FitResNet [177]	ICLR 2016	[code]	2.5M 2.5M	CIFAR-10 CIFAR-100	94.16 72.34
	ResInResNet [238]	ICLR 2016	[code]	10.3M 10.3M	CIFAR-10 CIFAR-100	94.99 77.10
	ResNeXt [262]	CVPR 2017	[code]	34.4M 34.4M	CIFAR-10 CIFAR-100	96.35 82.23
	FractalNet [34]	ICLR 2017	[code]	1.71M 22.9M	CIFAR-10 CIFAR-100	94.78 77.51
	RKCNN [294]	Neural Comput. & Appl. 2023	-	1.93M 1.96M	CIFAR-10 CIFAR-100	95.52 79.31
High-order guided	DenseNet [103]	CVPR 2017	[code]	7.0M 27.2M	CIFAR-10 CIFAR-100	95.90 80.75
	DenseNet-BC [103]			0.8M 15.3M	CIFAR-10 CIFAR-100	95.49 82.40
	LM-ResNets [164]	ICML 2018	[code]	1.14M 1.7M	CIFAR-10 CIFAR-100	93.84 74.13
	LM-ResNeXt [164]			- 35.1M	CIFAR-10 CIFAR-100	- 82.51
DEs-guided	RevNets [74]	NIPs 2017	[code]	1.75M 1.79M	CIFAR-10 CIFAR-100	94.83 86.31
	i-Resnets [12]	ICML 2019	[code]	8.7M -	CIFAR-10 CIFAR-100	96.55 -
	i-DenseNet [199]	NIPs 2021	[code]	8.7M -	CIFAR-10 CIFAR-100	96.63 -
PDE-guided	Parabolic CNNs [211]	JMIV 2020	[code]	0.26M 0.65M	CIFAR-10 CIFAR-100	88.5 64.8
	Hamiltonian CNNs [211]			0.51M	CIFAR-10	89.3
	second-order CNNs [211]			0.36M 0.51M	CIFAR-100 CIFAR-10	64.9 89.2
	Parabolic CNNs [211]			0.65M	CIFAR-100	65.4
	LeanResNets [57]	IEEE J-STSP 2020	-	0.66M -	CIFAR-10 CIFAR-100	93.4 -
	Parabolic PDE [212]	Program. Comput. Softw. 2023	-	11.17M -	CIFAR-10 CIFAR-100	95.42 -
	Hyperbolic PDE [212]			11.17M -	CIFAR-10 CIFAR-100	95.70 -

models like EDM achieve an FID of 1.97 with only 35 steps, and DEIS achieves 2.55 with 50 steps, demonstrating the importance of efficient numerical methods for practical generative modeling.

Overall, the transition from high-step stochastic sampling to single-step deterministic trajectory optimization has led to significant improvements in both generation quality and efficiency. ODE-guided models have established new benchmarks for fast inference, while SDE-guided models remain competitive in sample diversity and fidelity. Future progress will rely on the continued integration of mathematical innovation and efficient solver design to advance deep generative models toward broader and more scalable applications.

Table 6. Performance comparison of representative ODE-driven and SDE-driven generative models on CIFAR-10.

Flow-based models	Model	Publication	Code	Params	Dataset	FID↓	NFE↓	
Generalized Flow matching (Deterministic)	flow matching [147]	NIPs 2019	[code]	-	CIFAR-10	8.06	183	
	CFM [147]	NIPs 2019	[code]	-	CIFAR-10	6.35	142	
	DFM [69]	NIPs 2024	[code]	113M	CIFAR-10	5.04	1000	
	Generator Matching [99]	ICLR 2025	[code]	-	CIFAR-10	2.36	1000	
Trajectory Optimization (Deterministic)	Rectified Flow [152, 154]	ICLR 2023	[code]	61.8M	CIFAR-10	4.85	1	
	SlimFlow [289]	ECCV 2024	[code]	15.7M	CIFAR-10	5.02	1	
	Improving Recitified Flow [136]	NIPs 2024	[code]	-	CIFAR-10	3.07	1	
	Hierarchical Recitified Flow [289]	ICLR 2025	[code]	-	CIFAR-10	3.71	500	
Stochastic Flow-based models	MeanFlow [70]	-	[code]	55M	CIFAR-10	2.92	1	
	Forward SDE (Stochastic)	Model	Publication	Code	Params	Dataset	FID↓	NFE↓
	NCSN [228]	NIPs 2019	[code]	-	CIFAR-10	25.32	1000	
	NCSNv2 [229]	NIPs 2020	[code]	-	CIFAR-10	10.87	1160	
	DDPMs [97]	NIPs 2020	[code]	35.7M	CIFAR-10	3.21	1000	
	FastDPM [127]	ICML 2021	[code]	-	CIFAR-10	3.01	100	
	EDM [117]	NIPs 2022	[code]	55.7M	CIFAR-10	1.97	35	
	NI-DDPMs [249]	NIPs 2022 workshop	-	-	CIFAR-10	6.95	1000	
Reverse SDE (Stochastic)	LIM [274]	NIPs 2023	[code]	-	CIFAR-10	1.58	1000	
	Reflected SDE [160]	ICML 2023	[code]	35.7M	CIFAR-10	2.72	35	
	FP-diffusion [54]	ICML 2023	-	-	CIFAR-10	2.87	1000	
	DiffFlow [284]	NIPs 2021	[code]	-	CIFAR-10	13.43	1000	
Solvers for Reverse SDE (Stochastic)	Unified Framework [24]	NIPs 2023	[code]	-	CIFAR-10	6.7	1000	
	DEIS [285]	ICLR 2023	[code]	61.8M	CIFAR-10	2.55	50	
	ER-SDE [41]	WACV 2025	[code]	-	CIFAR-10	2.98	100	
	PF ODE [230]	ICLR 2021	[code]	35.7M	CIFAR-10	2.41	2000	
Other Solvers	DDIM [227]	ICLR 2021	[code]	35.7M	CIFAR-10	4.16	100	
	gDDIM [286]	ICLR 2023	[code]	-	CIFAR-10	2.26	50	
	DPM-Solver-2 [161]	NIPs 2022	[code]	35.7M	CIFAR-10	2.69	48	
	DPM-Solver-3 [161]	NIPs 2022	[code]	35.7M	CIFAR-10	2.65	48	
	PNDM [151]	ICLR 2022	[code]	-	CIFAR-10	3.26	1000	
	SA-Solver [265]	NIPs 2023	[code]	-	CIFAR-10	2.63	95	
	AMED-Solver [293]	CVPR 2024	[code]	55.7M	CIFAR-10	3.67	9	
	AMED-Plugin [293]	CVPR 2024	[code]	55.7M	CIFAR-10	2.63	9	

6 Discussions and Conclusion

6.1 Discussions

The integration of DEs with deep learning has opened up several promising avenues for advancing the field, but significant opportunities and challenges remain [100, 224, 240]. As the relationship between DEs and deep learning evolves, future research should focus on three key areas.

Firstly, DEs can guide the design of novel neural network architectures and regularization methods [23, 50, 164, 256]. Rather than limiting research to interpreting existing models, future efforts should actively investigate how different types of DEs can guide the creation of architectures that are better suited to specific tasks and demonstrate stronger generalization capabilities. To this end, it is essential to investigate a broader range of DE classes, including higher-order ODEs [17], delay differential equations [58], and fractional-order equations [30, 123], which can help models capture

complex dynamics such as memory effects and multi-scale behaviors. Furthermore, beyond the structure of the networks themselves, studying noise processes that go beyond traditional Gaussian assumptions, such as Lévy noise [200], could offer promising avenues for developing diverse and effective regularization strategies.

Then, DEs offer a unified and rigorous foundation for modeling dynamic systems in deep learning. Extending existing methods such as Neural ODEs [29] to broader formulations including Neural PDEs [36, 42] and Neural SDEs [122, 180] offers strong potential for capturing complex temporal dependencies and modeling richer system dynamics. Integrating ODEs and SDEs into generative modeling aids in learning flexible, high-fidelity data distributions. ODEs excel in interpretability and deterministic modeling [29, 84], while SDEs better express stochasticity and enhance generalization [155, 156].

In practice, ODEs and SDEs often complement each other: SDE frameworks are typically used during training to leverage noise perturbation and distribution coverage, while sampling allows for flexible use of SDE solvers (for diversity and quality) [22, 76, 106, 124, 138, 161, 162, 168, 186] or deterministic ODE solvers (for efficiency and reproducibility) to obtain a balanced generative performance and computational efficiency [14, 61, 148, 173].

Finally, applying DEs to the design of neural networks for specific real-world scenarios remains a critical area of research [39, 89, 102, 158, 181, 243, 270]. While significant progress has been made in integrating DEs into deep learning, their practical effectiveness often depends heavily on domain-specific requirements. For instance, solving physical problems involving partial PDEs typically demands high numerical precision [217] and computational stability. In contrast, computer vision tasks emphasize a model’s generalization capability, robustness, and interpretability [237]. In high-stakes applications such as healthcare [203] and autonomous driving [8, 81, 182], ensuring model reliability, accurate uncertainty quantification, and strict adherence to domain-specific constraints is particularly crucial [18, 219]. Therefore, future research should focus on more closely integrating the architectural design of neural networks with the mathematical properties of the corresponding DEs, to develop models that are both theoretically rigorous and practically effective.

To sum up, these emerging directions underscore the immense potential of DEs in driving innovation within deep learning, guiding the development of models that are both mathematically principled and highly effective across a wide range of applications.

6.2 Conclusion

In this survey, we provide an in-depth and comprehensive review of the intersection between deep learning and DEs, offering a broad perspective on the latest advancements and ongoing challenges in this dynamic research area. We begin by summarizing the progress in using DEs to interpret and design deep neural network architectures, as well as the role of SDEs in explaining and guiding stochastic regularization methods for deep neural networks. Additionally, we present a detailed overview of the progress in DE-guided deterministic and stochastic dynamical networks. Our aim is to equip researchers entering this field with a clear framework to facilitate their understanding of the diverse body of work and its theoretical foundations.

Furthermore, we examine the current state of research that combines DEs with deep learning, highlighting promising directions for future exploration and potential extensions. We identify opportunities for integrating more complex mathematical tools, extending these methods to broader application domains, and strengthening theoretical foundations to improve model interpretability and robustness. Due to space constraints, we provide concise summaries of key contributions rather than exhaustive technical details. We hope this survey serves as a valuable resource and inspires further research and innovation at the intersection of DEs and deep learning.

References

- [1] Abien Fred Agarap. 2018. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375* (2018).
- [2] Ravi Aggarwal, Viknesh Sounderajah, Guy Martin, Daniel SW Ting, Alan Karthikesalingam, Dominic King, Hutan Ashrafian, and Ara Darzi. 2021. Diagnostic accuracy of deep learning in medical imaging: a systematic review and meta-analysis. *NPJ digital medicine* 4, 1 (2021), 65.
- [3] Michael Samuel Albergo and Eric Vanden-Eijnden. 2023. Building Normalizing Flows with Stochastic Interpolants. In *International Conference on Learning Representations*.
- [4] Tobias Alt, Karl Schrader, Matthias Augustin, Pascal Peter, and Joachim Weickert. 2023. Connections between numerical algorithms for PDEs and neural networks. *Journal of Mathematical Imaging and Vision* 65, 1 (2023), 185–208.
- [5] Brian DO Anderson. 1982. Reverse-time diffusion equation models. *Stochastic Processes and their Applications* 12, 3 (1982), 313–326.
- [6] Ludwig Arnold. 1974. Stochastic differential equations. *New York* 2 (1974), 2.
- [7] Jimmy Ba and Brendan Frey. 2013. Adaptive dropout for training deep neural networks. In *Advances in Neural Information Processing Systems*, Vol. 26.
- [8] Bakary Badjie, Jose Cecilio, and Antonio Casimiro. 2024. Adversarial attacks and countermeasures on image classification-based deep learning models in autonomous driving systems: A systematic review. *Comput. Surveys* 57, 1 (2024), 1–52.
- [9] Pierre Baldi and Peter J Sadowski. 2013. Understanding dropout. In *Advances in Neural Information Processing Systems*, Vol. 26.
- [10] Chayan Banerjee, Kien Nguyen, Clinton Fookes, and Karniadakis George. 2024. Physics-Informed Computer Vision: A Review and Perspectives. *Comput. Surveys* 57, 1, Article 17 (Oct. 2024), 38 pages. doi:[10.1145/3689037](https://doi.org/10.1145/3689037)
- [11] Georgios Batzolos, Jan Stanczuk, Carola-Bibiane Schönlieb, and Christian Etmann. 2021. Conditional image generation with score-based diffusion models. *arXiv preprint arXiv:2111.13606* (2021).
- [12] Jens Behrmann, Will Grathwohl, Ricky TQ Chen, David Duvenaud, and Jörn-Henrik Jacobsen. 2019. Invertible residual networks. In *International Conference on Machine Learning*, Vol. 97. 573–582.
- [13] Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal. 2019. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences* 116, 32 (2019), 15849–15854. arXiv:<https://www.pnas.org/doi/pdf/10.1073/pnas.1903070116> doi:[10.1073/pnas.1903070116](https://doi.org/10.1073/pnas.1903070116)
- [14] Marin Biloš, Johanna Sommer, Syama Sundar Rangapuram, Tim Januschowski, and Stephan Günnemann. 2021. Neural flows: Efficient alternative to neural ODEs. In *Advances in Neural Information Processing Systems*, Vol. 34. 21325–21337.
- [15] Olivier Bousquet and Andre Elisseeff. 2002. Stability and generalization. *Journal of Machine Learning Research* 2 (2002), 499–526.
- [16] Martin Braun and Martin Golubitsky. 1983. *Differential equations and their applications*. Vol. 4. Springer.
- [17] John Charles Butcher. 2016. *Numerical methods for ordinary differential equations*. John Wiley & Sons.
- [18] Lei Cai, Jingyang Gao, and Di Zhao. 2020. A review of the application of deep learning in medical image classification and segmentation. *Annals of Translational Medicine* 8, 11 (2020), 713.
- [19] Andrew Campbell, Joe Benton, Valentin De Bortoli, Tom Rainforth, George Deligiannidis, and Arnaud Doucet. 2022. A continuous time framework for discrete denoising models. In *Advances in Neural Information Processing Systems*. 28266–28279.
- [20] Alexander Camuto. 2021. *Understanding gaussian noise injections in neural networks*. Ph.D. Dissertation. University of Oxford.
- [21] Alexander Camuto, Matthew Willetts, Umut Simsekli, Stephen J Roberts, and Chris C Holmes. 2020. Explicit regularisation in gaussian noise injections. In *Advances in Neural Information Processing Systems*, Vol. 33. 16603–16614.
- [22] Fei Cao, Kimball Johnston, Thomas Laurent, Justin Le, and Sébastien Motsch. 2025. Generative diffusion models from a PDE perspective. *arXiv preprint arXiv:2501.17054* (2025).
- [23] Hanqun Cao, Cheng Tan, Zhangyang Gao, Yilun Xu, Guangyong Chen, Pheng-Ann Heng, and Stan Z Li. 2024. A survey on generative diffusion models. *IEEE Transactions on Knowledge and Data Engineering* (2024).
- [24] Yu Cao, Jingrun Chen, Yixin Luo, and Xiang Zhou. 2023. Exploring the optimal choice for generative processes in diffusion models: Ordinary vs stochastic differential equations. In *Advances in Neural Information Processing Systems*, Vol. 36. 33420–33468.
- [25] Junyi Chai, Hao Zeng, Anning Li, and Eric W.T. Ngai. 2021. Deep learning in computer vision: A critical review of emerging techniques and application scenarios. *Machine Learning with Applications* 6 (2021), 100134. doi:[10.1016/j.mlwa.2021.100134](https://doi.org/10.1016/j.mlwa.2021.100134)
- [26] Bo Chang, Lili Meng, Eldad Haber, Lars Ruthotto, David Begert, and Elliot Holtham. 2018. Reversible architectures for arbitrarily deep residual neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 32.
- [27] Chunlei Chen, Peng Zhang, Huixiang Zhang, Jiangyan Dai, Yugen Yi, Huihui Zhang, and Yonghui Zhang. 2020. Deep Learning on Computational-Resource-Limited Platforms: A Survey. *Mobile Information Systems* 2020, 1 (2020), 8454327. arXiv:<https://onlinelibrary.wiley.com/doi/pdf/10.1155/2020/8454327> doi:[10.1155/2020/8454327](https://doi.org/10.1155/2020/8454327)
- [28] Ricky T. Q. Chen and Yaron Lipman. 2024. Flow Matching on General Geometries. In *International Conference on Learning Representations*.
- [29] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. 2018. Neural Ordinary Differential Equations. In *Advances in Neural Information Processing Systems*, Vol. 31.
- [30] YangQuan Chen, Ivo Petras, and Dingyu Xue. 2009. Fractional order control-a tutorial. In *2009 American control conference*. IEEE, 1397–1411.
- [31] Chaoran Cheng, Jiahan Li, Jiajun Fan, and Ge Liu. 2025. α -Flow: A Unified Framework for Continuous-State Discrete Flow Matching Models. *arXiv preprint arXiv:2504.10283* (2025).

- [32] Chaoran Cheng, Jiahao Li, Jian Peng, and Ge Liu. 2024. Categorical flow matching on statistical manifolds. In *Advances in Neural Information Processing Systems*, Vol. 37. 54787–54819.
- [33] Jer-Shiou Chiou and Yen-Hsien Lee. 2009. Jump dynamics and volatility: Oil and the stock markets. *Energy* 34, 6 (2009), 788–796. doi:10.1016/j.energy.2009.02.011
- [34] Woojin Cho, Seunghyeon Cho, Hyundong Jin, Jinsung Jeon, Kookjin Lee, Sanghyun Hong, Dongeun Lee, Jonghyun Choi, and Noseong Park. 2024. Operator-learning-inspired modeling of neural ordinary differential equations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 11543–11551.
- [35] Krzysztof M Choromanski, Jared Quincy Davis, Valerii Likhoshesterov, Xingyou Song, Jean-Jacques Slotine, Jacob Varley, Honglak Lee, Adrian Weller, and Vikas Sindhwani. 2020. Ode to an ODE. In *Advances in Neural Information Processing Systems*, Vol. 33. 3338–3350.
- [36] Cecilia Coelho, M Fernanda P Costa, and Luis F Ferrás. 2025. Neural fractional differential equations. *Applied Mathematical Modelling* 144 (2025), 116060.
- [37] Antoine Collas, Ce Ju, Nicolas Salvy, and Bertrand Thirion. 2025. Riemannian Flow Matching for Brain Connectivity Matrices via Pullback Geometry. *arXiv preprint arXiv:2505.18193* (2025).
- [38] Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: deep neural networks with multitask learning. In *International Conference on Machine Learning* (Helsinki, Finland). 160–167.
- [39] Florin-Alin Croitoru, Vlad Tudor Ionescu, and Mubarak Shah. 2023. Diffusion models in vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 9 (2023), 10850–10869.
- [40] Dominik Csiba and Peter Richtárik. 2018. Importance sampling for minibatches. *Journal of Machine Learning Research* 19, 27 (2018), 1–21.
- [41] Qinpeng Cui, Xinyi Zhang, Qiqi Bao, and Qingmin Liao. 2025. Elucidating the solution space of extended reverse-time SDE for diffusion models. In *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. 243–252.
- [42] Wenjun Cui, Qiyu Kang, Xuhua Li, Kai Zhao, Wee Peng Tay, Weihua Deng, and Yidong Li. 2025. Neural Variable-Order Fractional Differential Equation Networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 39. 16109–16117.
- [43] Wenjun Cui, Honglei Zhang, Haoyu Chu, Pipi Hu, and Yidong Li. 2023. On robustness of neural ODEs image classifiers. *Information Sciences* 632 (2023), 576–593.
- [44] Daems, Rembert and Opper, Manfred and Crevecoeur, Guillaume and Birdal, Tolga. 2025. Efficient training of neural SDEs using stochastic optimal control. In *ESANN 2025 : 33rd European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, Proceedings*. 693–698.
- [45] Quan Dao, Hao Phung, Binh Nguyen, and Anh Tran. 2023. Flow matching in latent space. *arXiv preprint arXiv:2307.08698* (2023).
- [46] Kalyan Das, Jiming Jiang, and JNK Rao. 2004. Mean squared error of empirical predictor. *The Annals of Statistics* 32, 2 (2004), 818–840.
- [47] Oscar Davis, Samuel Kessler, Mircea Petrache, Ismail Ceylan, Michael Bronstein, and Joey Bose. 2024. Fisher flow matching for generative modeling over discrete data. In *Advances in Neural Information Processing Systems*, Vol. 37. 139054–139084.
- [48] Arturo De Marinis, Nicola Guglielmi, Stefano Sicilia, and Francesco Tudisco. 2025. Stability of neural ODEs by a control over the expansivity of their flows. *arXiv preprint arXiv:2501.10740* (2025).
- [49] Li Deng and Yang Liu. 2018. *Deep learning in natural language processing*. Springer.
- [50] Teo Deveney, Jan Stanczuk, Lisa Kreusser, Chris Budd, and Carola-Bibiane Schönlieb. 2025. Closing the ODE–SDE gap in score-based diffusion models through the Fokker–Planck equation. *Philosophical Transactions A* 383, 2298 (2025), 20240503.
- [51] Omar Dhifallah and Yitong Lu. 2021. On the Inherent Regularization Effects of Noise Injection During Training. In *International Conference on Machine Learning*. 2676–2686.
- [52] Richard C Dorf. 1989. *Introduction to electric circuits*. John Wiley & Sons.
- [53] Finale Doshi-Velez and Been Kim. 2017. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608* (2017).
- [54] Weitao Du, He Zhang, Tao Yang, and Yuanqi Du. 2023. A flexible diffusion model. In *International Conference on Machine Learning*. 8678–8696.
- [55] Emilien Dupont, Arnaud Doucet, and Yee Whye Teh. 2019. Augmented neural odes. In *Advances in Neural Information Processing Systems*, Vol. 32.
- [56] Michael B Elowitz, Arnold J Levine, Eric D Siggia, and Peter S Swain. 2002. Stochastic gene expression in a single cell. *Science* 297, 5584 (2002), 1183–1186.
- [57] Jonathan Ephrath, Moshe Eliasof, Lars Ruthotto, Eldad Haber, and Eran Treister. 2020. LeanConvNets: low-cost yet effective convolutional neural networks. *IEEE Journal of Selected Topics in Signal Processing* 14, 4 (2020), 894–904.
- [58] Thomas Erneux. 2009. *Applied delay differential equations*. Springer.
- [59] Lawrence C Evans. 2022. *Partial differential equations*. Vol. 19. American Mathematical Society.
- [60] Angela Fan, Edouard Grave, and Armand Joulin. 2020. Reducing Transformer Depth on Demand with Structured Dropout. In *International Conference on Learning Representations*.
- [61] Chris Finlay, Jörn-Henrik Jacobsen, Levon Nurbekyan, and Adam Oberman. 2020. How to train your neural ODE: the world of Jacobian and kinetic regularization. In *International Conference on Machine Learning*. 3154–3164.
- [62] Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. 2025. One Step Diffusion via Shortcut Models. In *International Conference on Learning Representations*.
- [63] Peter K Friz and Martin Hairer. 2020. *A course on rough paths*. Springer.

- [64] Yuxiang Fu, Qi Yan, Lele Wang, Ke Li, and Renjie Liao. 2025. Moflow: One-step flow matching for human trajectory forecasting via implicit maximum likelihood estimation based distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 17282–17293.
- [65] Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. In *Advances in Neural Information Processing Systems*, Vol. 29.
- [66] Lucio Galeati and Fabian A Harang. 2022. Regularization of multiplicative SDEs through additive noise. *The Annals of Applied Probability* 32, 5 (2022), 3930–3963.
- [67] Crispin W Gardiner et al. 1985. *Handbook of stochastic methods*. Vol. 3. Springer Berlin.
- [68] Xavier Gastaldi. 2017. Shake-shake regularization of 3-branch residual networks. In *ICLR Workshop*.
- [69] Itai Gat, Tal Remez, Neta Shaul, Felix Kreuk, Ricky TQ Chen, Gabriel Synnaeve, Yossi Adi, and Yaron Lipman. 2024. Discrete flow matching. In *Advances in Neural Information Processing Systems*, Vol. 37. 133345–133385.
- [70] Zhengyang Geng, Mingyang Deng, Xingjian Bai, J Zico Kolter, and Kaiming He. 2025. Mean flows for one-step generative modeling. *arXiv preprint arXiv:2505.13447* (2025).
- [71] Golnaz Ghiasi, Tsung-Yi Lin, and Quoc V Le. 2018. Dropblock: A regularization method for convolutional networks. In *Advances in Neural Information Processing Systems*, Vol. 31.
- [72] Arnab Ghosh, Harkirat Behl, Emilien Dupont, Philip Torr, and Vinay Namboodiri. 2020. Steer: Simple temporal regularization for neural ode. In *Advances in Neural Information Processing Systems*, Vol. 33. 14831–14843.
- [73] Patryk Gierjatowicz, Marc Sabate-Vidales, David Siška, Lukasz Szpruch, and Žan Žurić. 2020. Robust pricing and hedging via neural SDEs. *arXiv preprint arXiv:2007.04154* (2020).
- [74] Aidan N Gomez, Mengye Ren, Raquel Urtasun, and Roger B Grosse. 2017. The reversible residual network: Backpropagation without storing activations. In *Advances in Neural Information Processing Systems*, Vol. 30.
- [75] Aidan N Gomez, Ivan Zhang, Siddhartha Rao Kamalakara, Divyam Madaan, Kevin Swersky, Yarin Gal, and Geoffrey E Hinton. 2019. Learning sparse networks using targeted dropout. *arXiv preprint arXiv:1905.13678* (2019).
- [76] Martin Gonzalez, Nelson Fernandez, Thuy Vinh Dinh Tran, Elies Gherbi, Hatem Hajri, and Nader Masmoudi. 2023. SEEDS: Exponential SDE Solvers for Fast High-Quality Sampling from Diffusion Models. In *Neural Information Processing Systems*.
- [77] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep learning*. MIT press.
- [78] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in Neural Information Processing Systems*. 2672–2680.
- [79] HS Greenside and E Helfand. 1981. Numerical integration of stochastic differential equations—II. *Bell System Technical Journal* 60, 8 (1981), 1927–1940.
- [80] Samuel Greydanus, Misko Dzamba, and Jason Yosinski. 2019. Hamiltonian neural networks. In *Advances in Neural Information Processing Systems*, Vol. 32.
- [81] Sorin Grigorescu, Bogdan Trasnea, Tiberiu Cocias, and Gigel Macesanu. 2020. A survey of deep learning techniques for autonomous driving. *Journal of Field Robotics* 37, 3 (2020), 362–386. arXiv:<https://onlinelibrary.wiley.com/doi/10.1002/rob.21918> doi:10.1002/rob.21918
- [82] Pengsheng Guo and Alex Schwing. 2025. Variational Rectified Flow Matching. In *International Conference on Machine Learning*.
- [83] Yuan Guo, Jingyu Kong, Yu Wang, and Yuping Duan. 2025. Take the Bull by the Horns: Learning to Segment Hard Samples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 15642–15652.
- [84] Abhinav Gupta and Pierre FJ Lermusiaux. 2023. Generalized neural closure models with interpretability. *Scientific Reports* 13, 1 (2023), 10634.
- [85] Eldad Haber and Lars Ruthotto. 2017. Stable architectures for deep neural networks. *Inverse problems* 34, 1 (2017), 014004.
- [86] Jun Han and Claudio Moraga. 1995. The influence of the sigmoid function parameters on the speed of backpropagation learning. In *From Natural to Artificial Neural Computation*. 195–201.
- [87] Sicong Han, Chenhao Lin, Chao Shen, Qian Wang, and Xiaohong Guan. 2023. Interpreting Adversarial Examples in Deep Learning: A Review. *Comput. Surveys* 55, 14s, Article 328 (July 2023), 38 pages. doi:10.1145/3594869
- [88] Soufiane Hayou and Fadhel Ayed. 2021. Regularization in resnet with stochastic depth. In *Advances in Neural Information Processing Systems*, Vol. 34. 15464–15474.
- [89] Chunming He, Yuqi Shen, Chengyu Fang, Fengyang Xiao, Longxiang Tang, Yulun Zhang, Wangmeng Zuo, Zhenhua Guo, and Xiu Li. 2025. Diffusion Models in Low-Level Vision: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 47, 6 (2025), 4630–4651.
- [90] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 770–778.
- [91] Xiangyu He, Zitao Mo, Peisong Wang, Yang Liu, Mingyuan Yang, and Jian Cheng. 2019. Ode-inspired network design for single image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 1732–1741.
- [92] Zhezhi He, Adnan Siraj Rakin, and Deliang Fan. 2019. Parametric noise injection: Trainable randomness to improve deep neural network robustness against adversarial attack. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 588–597.
- [93] Calypso Herrera, Florian Krach, and Josef Teichmann. 2021. Neural Jump Ordinary Differential Equations: Consistent Continuous-Time Prediction and Filtering. In *International Conference on Learning Representations*.
- [94] Steven L Heston. 1993. A closed-form solution for options with stochastic volatility with applications to bond and currency options. *The review of financial studies* 6, 2 (1993), 327–343.

- [95] David Hilbert. 1985. *Methods of mathematical physics*. CUP Archive.
- [96] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580* (2012).
- [97] Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, Vol. 33. 6840–6851.
- [98] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9, 8 (1997), 1735–1780.
- [99] Peter Holderrieth, Marton Havasi, Jason Yim, Neta Shaul, Itai Gat, Tommi Jaakkola, Brian Karrer, Ricky T. Q. Chen, and Yaron Lipman. 2025. Generator Matching: Generative modeling with arbitrary Markov processes. In *International Conference on Learning Representations*.
- [100] Md Tanzib Hosain, Jamin Rahman Jim, MF Mridha, and Md Mohsin Kabir. 2024. Explainable AI approaches in deep learning: Advancements, applications and challenges. *Computers and Electrical Engineering* 117 (2024), 109246.
- [101] Xiaoyang Hou, Tian Zhu, Milong Ren, Dongbo Bu, Xin Gao, Chunming Zhang, and Shiwei Sun. 2024. Improving Molecular Graph Generation with Flow Matching and Optimal Transport. In *NeurIPS 2024 Workshop on AI for New Drug Modalities*.
- [102] Xing Hua, Haodong Chen, Qianqian Duan, Danfeng Hong, Ruijiao Li, Huiliang Shang, Linghua Jiang, Haima Yang, and Dawei Zhang. 2025. A Comprehensive Review of Diffusion Models in Smart Agriculture: Progress, Applications, and Challenges. *arXiv preprint arXiv:2507.18376* (2025).
- [103] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. 2017. Densely connected convolutional networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 4700–4708.
- [104] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. 2016. Deep networks with stochastic depth. In *European Conference on Computer Vision*. 646–661.
- [105] Xingchang Huang, Corentin Salaun, Cristina Vasconcelos, Christian Theobalt, Cengiz Oztireli, and Gurprit Singh. 2024. Blue noise for diffusion models. In *ACM SIGGRAPH 2024 conference papers*. 1–11.
- [106] Yichuan Huang. 2025. Model Selection for Diffusion Coefficient Estimation in SDE. (Jan. 2025). <https://hal.science/hal-04901917> This paper focuses on the estimation of diffusion coefficients in stochastic differential equations using high-frequency data..
- [107] Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*. 448–456.
- [108] Zacharia Issa, Blanka Horvath, Maud Lemercier, and Cristopher Salvi. 2023. Non-adversarial training of Neural SDEs with signature kernel scores. In *Advances in Neural Information Processing Systems*, Vol. 36. 11102–11126.
- [109] Jörn-Henrik Jacobsen, Arnold Smeulders, and Edouard Oyallon. 2018. i-RevNet: Deep Invertible Networks. In *International Conference on Learning Representations*.
- [110] Arnulf Jentzen and Michael Röckner. 2015. A Milstein scheme for SPDEs. *Foundations of Computational Mathematics* 15 (2015), 313–362.
- [111] Junting Jia and Austin R Benson. 2019. Neural jump stochastic differential equations. In *Advances in Neural Information Processing Systems*, Vol. 32.
- [112] Ahmed Joudal, Youness El Moutaouakil, Ahmed El Hassani, and Youness El Moutaouakil. 2022. An Adaptive Drop Method for Deep Neural Networks Regularization. *Knowledge-Based Systems* 241 (2022), 108011.
- [113] Anil Kag and Venkatesh Saligrama. 2022. Condensing cnns with partial differential equations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 610–619.
- [114] Guoliang Kang, Jun Li, and Dacheng Tao. 2017. Shakeout: A new approach to regularized deep neural network training. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40, 5 (2017), 1245–1258.
- [115] Qiyu Kang, Yang Song, Qin Xu Ding, and Wee Peng Tay. 2021. Stable neural ode with lyapunov-stable equilibrium points for defending against adversarial attacks. In *Advances in Neural Information Processing Systems*, Vol. 34. 14925–14937.
- [116] Kacper Kapuńiak, Peter Potapchik, Teodora Reu, Leo Zhang, Alexander Tong, Michael Bronstein, Avishhek Joey Bose, and Francesco Di Giovanni. 2025. Metric flow matching for smooth interpolations on the data manifold. In *Neural Information Processing Systems*. Article 4291, 32 pages.
- [117] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. 2022. Elucidating the Design Space of Diffusion-Based Generative Models. In *Advances in Neural Information Processing Systems*, Vol. 35. 26565–26577.
- [118] Jacob Kelly, Jesse Bettencourt, Matthew J Johnson, and David K Duvenaud. 2020. Learning differential equations that are easy to solve. In *Advances in Neural Information Processing Systems*, Vol. 33. 4370–4380.
- [119] P Kidger. 2021. *On neural differential equations*. Ph.D. Dissertation. University of Oxford.
- [120] Patrick Kidger, James Foster, Xuechen Li, and Terry J Lyons. 2021. Neural sdes as infinite-dimensional gans. In *International Conference on Machine Learning*, Vol. 139. 5453–5463.
- [121] Patrick Kidger, James Foster, Xuechen Chen Li, and Terry Lyons. 2021. Efficient and accurate gradients for neural sdes. In *Advances in Neural Information Processing Systems*, Vol. 34. 18747–18761.
- [122] Patrick Kidger, James Morrill, James Foster, and Terry Lyons. 2020. Neural controlled differential equations for irregular time series. In *Advances in Neural Information Processing Systems*, Vol. 33. 6696–6707.
- [123] AA Kilbas and JJ Trujillo. 2001. Differential equations of fractional order: methods results and problem—I. *Applicable Analysis* 78, 1-2 (2001), 153–192.
- [124] Jongseon Kim, Hyungjoon Kim, HyunGi Kim, Dongjun Lee, and Sungroh Yoon. 2025. A comprehensive survey of deep learning for time series forecasting: architectural diversity and open challenges. *Artificial Intelligence Review* 58, 7 (2025), 1–95.

- [125] Durk P Kingma, Tim Salimans, and Max Welling. 2015. Variational dropout and the local reparameterization trick. In *Advances in Neural Information Processing Systems*, Vol. 28.
- [126] Lin Kong, Wei Sun, Fanhua Shang, Yuanyuan Liu, and Hongying Liu. 2022. HNO: High-Order Numerical Architecture for ODE-Inspired Deep Unfolding Networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 7220–7228.
- [127] Zhifeng Kong and Wei Ping. 2021. On Fast Sampling of Diffusion Probabilistic Models. In *ICML Workshop on Invertible Neural Networks, Normalizing Flows, and Explicit Likelihood Models*.
- [128] Nikita Kornilov, Petr Mokrov, Alexander Gasnikov, and Aleksandr Korotin. 2024. Optimal flow matching: Learning straight trajectories in just one step. In *Advances in Neural Information Processing Systems*, Vol. 37. 104180–104204.
- [129] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*. 1097–1105.
- [130] Anders Krogh and John A Hertz. 1992. A simple weight decay can improve generalization. In *Advances in Neural Information Processing Systems*, Vol. 4.
- [131] Siddharth Krishna Kumar. 2017. On weight initialization in deep neural networks. *arXiv preprint arXiv:1704.08863* (2017).
- [132] Alex Labach, Hojjat Salehinejad, and Shahrokh Valaee. 2019. Survey of dropout methods for deep neural networks. *arXiv preprint arXiv:1904.13310* (2019).
- [133] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. 2017. FractalNet: Ultra-Deep Neural Networks without Residuals. In *International Conference on Learning Representations*.
- [134] Quoc V Le, Jiquan Ngiam, Adam Coates, Abhik Lahiri, Bobby Prochnow, and Andrew Y Ng. 2011. On optimization methods for deep learning. In *International Conference on Machine Learning*. 265–272.
- [135] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature* 521, 7553 (2015), 436–444.
- [136] Sangyun Lee, Zinan Lin, and Giulia Fanti. 2024. Improving the training of rectified flows. In *Advances in Neural Information Processing Systems*, Vol. 37. 63082–63109.
- [137] Mikko Lehtimäki, Lassi Paunonen, and Marja-Leena Linne. 2022. Accelerating neural odes using model order reduction. *IEEE Transactions on Neural Networks and Learning Systems* 35, 1 (2022), 519–531.
- [138] Ao Li, Wei Fang, Hongbo Zhao, Le Lu, Ge Yang, and Minfeng Xu. 2025. MaRS: A Fast Sampler for Mean Reverting Diffusion based on ODE and SDE Solvers. *arXiv preprint arXiv:2502.07856* (2025).
- [139] Mufan Li, Mihai Nica, and Dan Roy. 2022. The neural covariance SDE: Shaped infinite depth-and-width networks at initialization. In *Advances in Neural Information Processing Systems*, Vol. 35. 10795–10808.
- [140] Ruichuan Li, Qiyou Sun, Xinkai Ding, Yisheng Zhang, Wentao Yuan, and Tong Wu. 2022. Review of Flow-Matching Technology for Hydraulic Systems. *Processes* 10, 12 (2022). doi:10.3390/pr10122482
- [141] Shufan Li, Konstantinos Kallidromitis, Akash Goluk, Zichun Liao, Yusuke Kato, Kazuki Kozuka, and Aditya Grover. 2025. Omnidflow: Any-to-any generation with multi-modal rectified flows. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 13178–13188.
- [142] Shan Li and Yiu-Ming Liu. 2016. Whiteout: Gaussian adaptive noise regularization in deep neural networks. *arXiv preprint arXiv:1612.01490* (2016).
- [143] Xuechen Li, Ting-Kam Leonard Wong, Ricky TQ Chen, and David Duvenaud. 2020. Scalable gradients for stochastic differential equations. In *International Conference on Artificial Intelligence and Statistics*, Vol. 108. 3870–3882.
- [144] Zifeng Lian, Xiaojun Jing, Xiaohan Wang, Hai Huang, Youheng Tan, and Yuanhao Cui. 2016. DropConnect regularization method with sparsity constraint for neural networks. *Chinese Journal of Electronics* 25, 1 (2016), 152–158.
- [145] Kang Liao, Zongsheng Yue, Zhouxia Wang, and Chen Change Loy. [n. d.]. Denoising as Adaptation: Noise-Space Domain Adaptation for Image Restoration. In *The Thirteenth International Conference on Learning Representations*.
- [146] Alec J Linot, Joshua W Burby, Qi Tang, Prasanna Balaprakash, Michael D Graham, and Romit Maulik. 2023. Stabilized neural ordinary differential equations for long-time forecasting of dynamical systems. *J. Comput. Phys.* 474 (2023), 111838.
- [147] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. 2023. Flow matching for generative modeling. In *International Conference on Learning Representations*.
- [148] Yaron Lipman, Marton Havasi, Peter Holderrieth, Neta Shaul, Matt Le, Brian Karrer, Ricky TQ Chen, David Lopez-Paz, Heli Ben-Hamu, and Itai Gat. 2024. Flow matching guide and code. *arXiv preprint arXiv:2412.06264* (2024).
- [149] Enshu Liu, Xuefei Ning, Yu Wang, and Zinan Lin. 2025. Distilled Decoding 1: One-step Sampling of Image Auto-regressive Models with Flow Matching. In *The Thirteenth International Conference on Learning Representations*. <https://openreview.net/forum?id=zKIFXV87Pp>
- [150] Guan-Horng Liu, Tianrong Chen, and Evangelos Theodorou. 2021. Second-order neural ode optimizer. In *Advances in Neural Information Processing Systems*, Vol. 34. 25267–25279.
- [151] Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao. 2022. Pseudo Numerical Methods for Diffusion Models on Manifolds. In *International Conference on Learning Representations*.
- [152] Qiang Liu. 2022. Rectified flow: A marginal preserving approach to optimal transport. *arXiv preprint arXiv:2209.14577* (2022).
- [153] Weibo Liu, Zidong Wang, Xiaohui Liu, Nianyin Zeng, Yurong Liu, and Fuad E. Alsaadi. 2017. A survey of deep neural network architectures and their applications. *Neurocomputing* 234 (2017), 11–26. doi:10.1016/j.neucom.2016.12.038
- [154] Xingchao Liu, Chengyue Gong, and qiang liu. 2023. Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow. In *International Conference on Learning Representations*.

- [155] Xuanqing Liu, Tesi Xiao, Si Si, Qin Cao, Sanjiv Kumar, and Cho-Jui Hsieh. 2019. Neural sde: Stabilizing neural ode networks with stochastic noise. *arXiv preprint arXiv:1906.02355* (2019).
- [156] Xuanqing Liu, Tesi Xiao, Si Si, Qin Cao, Sanjiv Kumar, and Cho-Jui Hsieh. 2020. How does noise help robustness? explanation and exploration under the neural sde framework. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 282–290.
- [157] Xingchao Liu, Xiwen Zhang, Jianzhu Ma, Jian Peng, et al. 2023. Instaflow: One step is enough for high-quality diffusion-based text-to-image generation. In *International Conference on Learning Representations*.
- [158] Yan Liu, Tao Jiang, Rui Li, Lingling Yuan, Marcin Grzegorzek, Chen Li, and Xiaoyan Li. 2025. A state-of-the-art review of diffusion model applications for microscopic image and micro-alike image analysis. *Frontiers in Medicine* 12 (2025), 1551894.
- [159] Andreas Look, Melih Kandemir, Barbara Rakitsch, and Jan Peters. 2022. A deterministic approximation to neural SDEs. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 4 (2022), 4023–4037.
- [160] Aaron Lou and Stefano Ermon. 2023. Reflected diffusion models. In *International Conference on Machine Learning*. 22675–22701.
- [161] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. 2022. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. In *Advances in Neural Information Processing Systems*, Vol. 35. 5775–5787.
- [162] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. 2025. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *Machine Intelligence Research* (2025), 1–22.
- [163] Siqi Lu, Fengyu Guan, Hanyu Zhang, and Haitao Lai. 2023. Speed-up ddpm for real-time underwater image enhancement. *IEEE Transactions on Circuits and Systems for Video Technology* 34, 5 (2023), 3576–3588.
- [164] Yiping Lu, Aoxiao Zhong, Quanzheng Li, and Bin Dong. 2018. Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations. In *International Conference on Machine Learning*, Vol. 80. 3276–3285.
- [165] Ziwei Luo, Fredrik K Gustafsson, Zheng Zhao, Jens Sjölund, and Thomas B Schön. 2023. Image restoration with mean-reverting stochastic differential equations. In *International Conference on Machine Learning*. 23045–23066.
- [166] Zhengbo Luo, Zitang Sun, Weilian Zhou, Zizhang Wu, and Sei-ichiro Kamata. 2022. Rethinking ResNets: improved stacking strategies with high-order schemes for image classification. *Complex & Intelligent Systems* 8, 4 (2022), 3395–3407.
- [167] Zhiyuan Ma, Ruixin Liu, Sixian Liu, Jianjun Li, and Bowen Zhou. 2025. Flow Diverse and Efficient: Learning Momentum Flow Matching via Stochastic Velocity Field Sampling. *arXiv preprint arXiv:2506.08796* (2025).
- [168] Zhiyuan Ma, Yuzhu Zhang, Guoli Jia, Liangliang Zhao, Yichao Ma, Mingjie Ma, Gaofeng Liu, Kaiyan Zhang, Ning Ding, Jianjun Li, et al. 2025. Efficient diffusion models: A comprehensive survey from principles to practices. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2025).
- [169] Shie Mannor, Dori Peleg, and Reuven Rubinstein. 2005. The cross entropy method for classification. In *International Conference on Machine Learning*. 561–568.
- [170] Andre Martins and Ramon Astudillo. 2016. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *International Conference on Machine Learning*. 1614–1623.
- [171] Gisiro Maruyama. 1955. Continuous Markov processes and stochastic equations. *Rendiconti del Circolo Matematico di Palermo* 4 (1955), 48–90.
- [172] Stefano Massaroli, Michael Poli, Michelangelo Bin, Jinkyoo Park, Atsushi Yamashita, and Hajime Asama. 2020. Stable neural flows. *arXiv preprint arXiv:2003.08063* (2020).
- [173] Stefano Massaroli, Michael Poli, Jinkyoo Park, Atsushi Yamashita, and Hajime Asama. 2020. Dissecting neural odes. In *Advances in Neural Information Processing Systems*, Vol. 33. 3952–3963.
- [174] Sam McCallum and James Foster. 2024. Efficient, accurate and stable gradients for neural odes. *arXiv preprint arXiv:2410.11648* (2024).
- [175] Brett A Melbourne and Alan Hastings. 2008. Extinction risk depends strongly on factors contributing to stochasticity. *Nature* 454, 7200 (2008), 100–103.
- [176] Grigorii Noikhovich Milstein. 2013. *Numerical integration of stochastic differential equations*. Vol. 313. Springer Science & Business Media.
- [177] Dmytro Mishkin and Jiri Matas. 2015. All you need is a good init. *arXiv preprint arXiv:1511.06422* (2015).
- [178] Reza Moradi, Reza Berangi, and Behrouz Minaei. 2020. A survey of regularization strategies for deep models. *Artificial Intelligence Review* 53, 6 (2020), 3947–3986.
- [179] Pietro Moretto, Jacopo Cavazza, Riccardo Volpi, René Vidal, and Vittorio Murino. 2017. Curriculum dropout. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 3544–3552.
- [180] James Morrill, Christopher Salvi, Patrick Kidger, and James Foster. 2021. Neural rough differential equations for long time series. In *International Conference on Machine Learning*. 7829–7838.
- [181] Brian B. Moser, Arundhati S. Shanbhag, Federico Raue, Stanislav Frolov, Sebastian Palacio, and Andreas Dengel. 2025. Diffusion Models, Image Super-Resolution, and Everything: A Survey. *IEEE Transactions on Neural Networks and Learning Systems* 36, 7 (2025), 11793–11813. doi:10.1109/TNNLS.2024.3476671
- [182] Khan Muhammad, Amin Ullah, Jaime Lloret, Javier Del Ser, and Victor Hugo C. de Albuquerque. 2021. Deep Learning for Safe Autonomous Driving: Current Challenges and Future Directions. *IEEE Transactions on Intelligent Transportation Systems* 22, 7 (2021), 4316–4336. doi:10.1109/TITS.2020.3032227
- [183] James D Murray. 2007. *Mathematical biology: I. An introduction*. Vol. 17. Springer Science & Business Media.

- [184] Meenal V Narkhede, Prashant P Bartakke, and Mukul S Sutaone. 2022. A review on weight initialization strategies for neural networks. *Artificial Intelligence Review* 55, 1 (2022), 291–322.
- [185] Ho Huu Nghia Nguyen, Tan Nguyen, Huyen Vo, Stanley Osher, and Thieu Vo. 2022. Improving neural ordinary differential equations with nesterov's accelerated gradient method. In *Advances in Neural Information Processing Systems*, Vol. 35. 7712–7726.
- [186] Shen Nie, Hanzhong Allan Guo, Cheng Lu, Yuhao Zhou, Chenyu Zheng, and Chongxuan Li. 2024. The blessing of randomness: Sde beats ode in general diffusion-based image editing. In *International Conference on Learning Representations*.
- [187] Hao Niu, Yuxiang Zhou, Xiaohao Yan, Jun Wu, Yuncheng Shen, Zhang Yi, and Junjie Hu. 2024. On the applications of neural ordinary differential equations in medical image analysis. *Artificial Intelligence Review* 57, 9 (2024), 236.
- [188] Hyeyoung Noh, Tackgeun You, Jonghwan Mun, and Bohyung Han. 2017. Regularizing deep neural networks by noise: Its interpretation and optimization. In *Advances in Neural Information Processing Systems*, Vol. 30.
- [189] Alexander Norcliffe, Cristian Bodnar, Ben Day, Jacob Moss, and Pietro Liò. 2021. Neural ODE Processes. In *International Conference on Learning Representations*.
- [190] Alexander Norcliffe, Cristian Bodnar, Ben Day, Nikola Simidjevski, and Pietro Liò. 2020. On second order behaviour in augmented neural odes. In *Advances in Neural Information Processing Systems*, Vol. 33. 5911–5921.
- [191] YongKyung Oh, Seungsu Kam, Jonghun Lee, Dong-Young Lim, Sungil Kim, and Alex Bui. 2025. Comprehensive review of neural differential equations for time series analysis. *arXiv preprint arXiv:2502.09885* (2025).
- [192] YongKyung Oh, Dongyoung Lim, and Sungil Kim. 2024. Stable Neural Stochastic Differential Equations in Analyzing Irregular Time Series Data. In *International Conference on Learning Representations*.
- [193] Bernt Oksendal. 2013. *Stochastic differential equations: an introduction with applications*. Springer Science & Business Media.
- [194] Antonio Orvieto, Anant Raj, Hans Kersting, and Francis Bach. 2023. Explicit regularization in overparametrized models via noise injection. In *International Conference on Artificial Intelligence and Statistics*. 7265–7287.
- [195] Daniel W. Otter, Julian R. Medina, and Jugal K. Kalita. 2021. A Survey of the Usages of Deep Learning for Natural Language Processing. *IEEE Transactions on Neural Networks and Learning Systems* 32, 2 (2021), 604–624. doi:10.1109/TNNLS.2020.2979670
- [196] Ambar Pal, Connor Lane, René Vidal, and Benjamin D Haefele. 2020. On the regularization properties of structured dropout. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 7671–7679.
- [197] Dogyun Park, Sojin Lee, Siheyon Kim, Taehoon Lee, Youngjoon Hong, and Hyunwoo J. Kim. 2024. Constant Acceleration Flow. In *Advances in Neural Information Processing Systems*, Vol. 37. 90030–90060.
- [198] Sung Woo Park, Hyomin Kim, Kyungjae Lee, and Junseok Kwon. 2022. Riemannian neural SDE: learning stochastic representations on manifolds. In *Advances in Neural Information Processing Systems*, Vol. 35. 1434–1444.
- [199] Yura Perugachi-Diaz, Jakub Tomczak, and Sandjai Bhulai. 2021. Invertible densenets with concatenated lipswish. In *Advances in Neural Information Processing Systems*, Vol. 34. 17246–17257.
- [200] Szymon Peszat and Jerzy Zabczyk. 2007. *Stochastic partial differential equations with Lévy noise: An evolution equation approach*. Vol. 113. Cambridge University Press.
- [201] Tomaso Poggio, Hrushikesh Mhaskar, Lorenzo Rosasco, Brando Miranda, and Qianli Liao. 2017. Why and when can deep-but not shallow-networks avoid the curse of dimensionality: a review. *International Journal of Automation and Computing* 14, 5 (2017), 503–519.
- [202] Lev Semenovich Pontryagin. 2018. *Mathematical theory of optimal processes*. Routledge.
- [203] Anichur Rahman, Tanoy Debnath, Dipanjali Kundu, Md Saikat Islam Khan, Airin Afroz Aishi, Sadia Sazzad, Mohammad Sayuzzaman, and Shahab S Band. 2024. Machine learning and deep learning-based approach in smart healthcare: Recent advances, applications, challenges and opportunities. *AIMS Public Health* 11, 1 (2024), 58. <https://www.aimspress.com/article/doi/10.3934/publichealth.2024004>
- [204] Steven J Rennie, Vaibhava Goel, and Samuel Thomas. 2014. Annealed dropout training of deep networks. In *2014 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 159–164.
- [205] Pierre Harvey Richemond, Sander Dieleman, and Arnaud Doucet. 2023. Categorical SDEs with Simplex Diffusion. In *ICML 2023 Workshop: Sampling and Optimization in Discrete Space*.
- [206] Hannes Risken. 1989. Fokker-planck equation. In *The Fokker-Planck equation: methods of solution and applications*. Springer, 63–95.
- [207] Hannes Risken. 1996. *Fokker-Planck Equation*. Springer Berlin Heidelberg, Berlin, Heidelberg, 63–95. doi:10.1007/978-3-642-61544-3_4
- [208] Ivan Dario Jimenez Rodriguez, Aaron Ames, and Yisong Yue. 2022. Lyanet: A lyapunov framework for training neural odes. In *International Conference on Machine Learning*. 18687–18703.
- [209] Yulia Rubanova, Ricky TQ Chen, and David K Duvenaud. 2019. Latent ordinary differential equations for irregularly-sampled time series. In *Advances in Neural Information Processing Systems*, Vol. 32.
- [210] Cynthia Rudin. 2019. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence* 1, 5 (2019), 206–215.
- [211] Lars Ruthotto and Eldad Haber. 2020. Deep neural networks motivated by partial differential equations. *Journal of Mathematical Imaging and Vision* 62, 3 (2020), 352–364.
- [212] Vardan R Sahakyan, Vahagn G Melkonyan, Gor A Gharagozyan, and Arman S Avetisyan. 2023. Enhancing Image Recognition with Pre-Defined Convolutional Layers Based on PDEs. *Programming and Computer Software* 49, 3 (2023), 192–197.

- [213] Imrus Salehin and Dae-Ki Kang. 2023. A review on dropout regularization approaches for deep neural networks within the scholarly domain. *Electronics* 12, 14 (2023), 3106. <https://www.mdpi.com/2079-9292/12/14/3106>
- [214] Hojjat Salehinejad and Shahrokh Valaee. 2021. Edropout: Energy-based dropout and pruning of deep neural networks. *IEEE Transactions on Neural Networks and Learning Systems* 33, 10 (2021), 5279–5292.
- [215] Cristopher Salvi, Maud Lemercier, and Andris Gerasimovics. 2022. Neural stochastic pdes: Resolution-invariant learning of continuous spatiotemporal dynamics. In *Advances in Neural Information Processing Systems*, Vol. 35. 1333–1344.
- [216] Johannes Schusterbauer, Ming Gui, Frank Fundel, and Björn Ommer. 2025. Diff2Flow: Training Flow Matching Models via Diffusion Model Alignment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 28347–28357.
- [217] Granville Sewell. 2005. *The numerical solution of ordinary and partial differential equations*. Vol. 75. John Wiley & Sons.
- [218] Dario Shariati, Umut Simsekli, and Alain Durmus. 2024. Denoising L¹evy Probabilistic Models. *arXiv preprint arXiv:2407.18609* (2024).
- [219] Dinggang Shen, Guorong Wu, and Heung-Il Suk. 2017. Deep learning in medical image analysis. *Annual review of biomedical engineering* 19, 1 (2017), 221–248.
- [220] Macheng Shen and Chen Cheng. 2025. Neural SDEs as a Unified Approach to Continuous-Domain Sequence Modeling. *arXiv preprint arXiv:2501.18871* (2025).
- [221] Cristina Silvano, Daniele Ielmini, Fabrizio Ferrandi, Leandro Fiorin, Serena Curzel, Luca Benini, Francesco Conti, Angelo Garofalo, Cristian Zambelli, Enrico Calore, Sebastiano Schifano, Maurizio Palesi, Giuseppe Ascia, Davide Patti, Nicola Petra, Davide De Caro, Luciano Lavagno, Teodoro Urso, Valeria Cardellini, Gian Carlo Cardarilli, Robert Birke, and Stefania Perri. 2025. A Survey on Deep Learning Hardware Accelerators for Heterogeneous HPC Platforms. *Comput. Surveys* 57, 11, Article 286 (June 2025), 39 pages. doi:10.1145/3729215
- [222] Saurabh Singh, Derek Hoiem, and David Forsyth. 2016. Swapout: Learning an ensemble of deep architectures. In *Advances in Neural Information Processing Systems*, Vol. 29.
- [223] Satya P. Singh, Lipo Wang, Sukrit Gupta, Haveesh Goli, Parasuraman Padmanabhan, and Balázs Gulyás. 2020. 3D Deep Learning on Medical Images: A Review. *Sensors* 20, 18 (2020). doi:10.3390/s20185097
- [224] Bart MN Smets, Jim Portegies, Erik J Bekkers, and Remco Duits. 2023. PDE-based group equivariant convolutional neural networks. *Journal of Mathematical Imaging and Vision* 65, 1 (2023), 209–239.
- [225] Luke Snow and Vikram Krishnamurthy. 2025. Efficient Neural SDE Training using Wiener-Space Cubature. *arXiv preprint arXiv:2502.12395* (2025).
- [226] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. 2015. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*. 2256–2265.
- [227] Jiaming Song, Chenlin Meng, and Stefano Ermon. 2021. Denoising Diffusion Implicit Models. In *International Conference on Learning Representations*.
- [228] Yang Song and Stefano Ermon. 2019. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems*, Vol. 32.
- [229] Yang Song and Stefano Ermon. 2020. Improved Techniques for Training Score-Based Generative Models. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates, Inc., 12438–12448.
- [230] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. 2021. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*.
- [231] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. 2021. Score-Based Generative Modeling through Stochastic Differential Equations. In *International Conference on Learning Representations*.
- [232] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research* 15, 1 (2014), 1929–1958.
- [233] Walter A Strauss. 2007. *Partial differential equations: An introduction*. John Wiley & Sons.
- [234] Qi Sun, Yunzhe Tao, and Qiang Du. 2018. Stochastic training of residual networks: a differential equation viewpoint. *arXiv preprint arXiv:1812.00174* (2018).
- [235] Ruo-Yu Sun. 2020. Optimization for deep learning: An overview. *Journal of the Operations Research Society of China* 8, 2 (2020), 249–294.
- [236] Vivienne Sze, Yu-Hsin Chen, Joel Emer, Amr Suleiman, and Zhengdong Zhang. 2017. Hardware for machine learning: Challenges and opportunities. In *2017 IEEE Custom Integrated Circuits Conference (CICC)*. 1–8. doi:10.1109/CICC.2017.7993626
- [237] Richard Szeliski. 2022. *Computer vision: algorithms and applications*. Springer Nature.
- [238] Sasha Targ, Diogo Almeida, and Kevin Lyman. 2016. Resnet in resnet: Generalizing residual architectures. *arXiv preprint arXiv:1603.08029* (2016).
- [239] Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, and Christoph Bregler. 2015. Efficient object localization using convolutional networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 648–656.
- [240] Maria Trigka and Elias Dritsas. 2025. A comprehensive survey of deep learning approaches in image processing. *Sensors* 25, 2 (2025), 531.
- [241] Aaron Tuor, Jan Drgona, and Draguna Vrabie. 2019. Constrained Neural Ordinary Differential Equations with Stability Guarantees. In *ICLR 2020 Workshop on Integration of Deep Neural Models and Differential Equations*.
- [242] Belinda Tzen and Maxim Raginsky. 2019. Neural stochastic differential equations: Deep latent gaussian models in the diffusion limit. *arXiv preprint arXiv:1905.09883* (2019).
- [243] Anwaar Ulhaq and Naveed Akhtar. 2022. Efficient diffusion models for vision: A survey. *arXiv preprint arXiv:2210.09292* (2022).
- [244] Hamit Taner Ünal and Fatih Başçiftçi. 2022. Evolutionary design of neural network architectures: a review of three decades of research. *Artificial Intelligence Review* 55, 3 (2022), 1723–1802.

- [245] Nicolaas G Van Kampen. 1976. Stochastic differential equations. *Physics Reports* 24, 3 (1976), 171–228.
- [246] Jente Vandersanden, Sascha Holl, Xingchang Huang, and Gurprit Singh. 2025. Edge-preserving noise for diffusion models. In *ICLR 2025 Workshop on Deep Generative Model in Machine Learning: Theory, Principle and Efficacy*.
- [247] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*. 5998–6008.
- [248] Andreas Veit, Neil Alldrin, Gal Chechik, Ivan Krasin, Abhinav Gupta, and Serge Belongie. 2017. Learning from noisy large-scale datasets with minimal supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 839–847.
- [249] Vikram Voleti, Christopher Pal, and Adam M Oberman. 2022. Score-based Denoising Diffusion with Non-Isotropic Gaussian Noise Models. In *NeurIPS 2022 Workshop on Score-Based Methods*.
- [250] Athanasios Voulodimos, Nikolaos Doulamis, Anastasios Doulamis, and Eftychios Protopapadakis. 2018. Deep Learning for Computer Vision: A Brief Review. *Computational Intelligence and Neuroscience* 2018, 1 (2018), 7068349. arXiv:<https://onlinelibrary.wiley.com/doi/pdf/10.1155/2018/7068349> doi:[10.1155/2018/7068349](https://doi.org/10.1155/2018/7068349)
- [251] Li Wan, Matthew Zeiler, Sixin Zhang, Yann Le Cun, and Rob Fergus. 2013. Regularization of neural networks using dropconnect. In *International Conference on Machine Learning*. 1058–1066.
- [252] Fu-Yun Wang, Ling Yang, Zhaoyang Huang, Mengdi Wang, and Hongsheng Li. 2025. Rectified Diffusion: Straightness Is Not Your Need in Rectified Flow. In *International Conference on Learning Representations*.
- [253] Luran Wang, Chaoran Cheng, Yizhen Liao, Yanru Qu, and Ge Liu. 2025. Training Free Guided Flow-Matching with Optimal Control. In *International Conference on Learning Representations*.
- [254] Shengbo Wang, Jose Blanchet, and Peter Glynn. 2024. An Efficient High-dimensional Gradient Estimator for Stochastic Differential Equations. In *Advances in Neural Information Processing Systems*, Vol. 37. 88045–88090.
- [255] Sida Wang and Christopher Manning. 2013. Fast dropout training. In *International Conference on Machine Learning*. 118–126.
- [256] Tangjun Wang, Chenglong Bao, and Zuoqiang Shi. 2025. Convection-Diffusion Equation: A Theoretically Certified Framework for Neural Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 47, 5 (2025), 4170–4182. doi:[10.1109/TPAMI.2025.3540310](https://doi.org/10.1109/TPAMI.2025.3540310)
- [257] Ee Weinan. 2017. A proposal on machine learning via dynamical systems. *Communications in Mathematics and Statistics* 5, 1 (2017), 1–11.
- [258] Bogdan M. Wilamowski. 2009. Neural network architectures and learning algorithms. *IEEE Industrial Electronics Magazine* 3, 4 (2009), 56–63. doi:[10.1109/MIE.2009.934790](https://doi.org/10.1109/MIE.2009.934790)
- [259] Haibing Wu and Xiaodong Gu. 2015. Max-pooling dropout for regularization of convolutional neural networks. In *Neural Information Processing: 22nd International Conference, ICONIP 2015, Istanbul, Turkey, November 9–12, 2015, Proceedings, Part I* 22. Springer, 46–54.
- [260] Yuchen Wu, Yuxin Chen, and Yuting Wei. 2024. Stochastic runge-kutta methods: Provable acceleration of diffusion models. *arXiv preprint arXiv:2410.04760* (2024).
- [261] Hedi Xia, Vai Suliafu, Hangjie Ji, Tan Nguyen, Andrea Bertozzi, Stanley Osher, and Bao Wang. 2021. Heavy ball neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, Vol. 34. 18646–18659.
- [262] Saining Xie, Ross Girshick, Piotr Dollar, Zhuowen Tu, and Kaiming He. 2017. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 1492–1500.
- [263] Wei-Chau Xie. 2010. *Differential equations for engineers*. Cambridge university press.
- [264] Kaiwen Xue, Yuhao Zhou, Shen Nie, Xu Min, Xiaolu Zhang, JUN ZHOU, and Chongxuan Li. 2024. Unifying Bayesian Flow Networks and Diffusion Models through Stochastic Differential Equations. In *International Conference on Machine Learning*.
- [265] Shuchen Xue, Mingyang Yi, Weijian Luo, Shifeng Zhang, Jiacheng Sun, Zhenguo Li, and Zhi-Ming Ma. 2023. Sa-solver: Stochastic adams solver for fast sampling of diffusion models. In *Advances in Neural Information Processing Systems*, Vol. 36. 77632–77674.
- [266] Yu Yamada, Masaki Iwamura, Takuya Akiba, and Koichi Kise. 2019. ShakeDrop regularization for deep residual learning. *IEEE Access* 7 (2019), 186126–186136.
- [267] Yoshihiro Yamada, Masakazu Iwamura, and Koichi Kise. 2016. Deep Pyramidal Residual Networks with Separated Stochastic Depth. *CoRR* abs/1612.01230 (2016). <http://arxiv.org/abs/1612.01230>
- [268] Hanshu YAN, Jiawei DU, Vincent TAN, and Jiashi FENG. 2020. On Robustness of Neural Ordinary Differential Equations. In *International Conference on Learning Representations*.
- [269] Hanshu Yan, Xingchao Liu, Jiaochun Pan, Jun Hao Liew, Qiang Liu, and Jiashi Feng. 2024. Perflow: Piecewise rectified flow as universal plug-and-play accelerator. In *Advances in Neural Information Processing Systems*, Vol. 37. 78630–78652.
- [270] Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. 2023. Diffusion models: A comprehensive survey of methods and applications. *Comput. Surveys* 56, 4 (2023), 1–39.
- [271] Ling Yang, Zixiang Zhang, Zhilong Zhang, Xingchao Liu, Minkai Xu, Wentao Zhang, Chenlin Meng, Stefano Ermon, and Bin Cui. 2024. Consistency Flow Matching: Defining Straight Flows with Velocity Consistency. *CoRR* abs/2407.02398 (2024). <https://doi.org/10.48550/arXiv.2407.02398>
- [272] Yuan-Chih Yang and Hung-Hsuan Chen. 2025. Dynamic DropConnect: Enhancing Neural Network Robustness Through Adaptive Edge Dropping Strategies. In *Data Science: Foundations and Applications*. 110–121.
- [273] Nanyang Ye, Linfeng Cao, Liujia Yang, Ziqing Zhang, Zhicheng Fang, Qinying Gu, and Guang-Zhong Yang. 2023. Improving the robustness of analog deep neural networks through a Bayes-optimized noise injection approach. *Communications Engineering* 2, 1 (2023), 25.

- [274] Eun Bi Yoon, Keehun Park, Sungwoong Kim, and Sungbin Lim. 2023. Score-based generative models with Lévy processes. In *Advances in Neural Information Processing Systems*, Vol. 36. 40694–40707.
- [275] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. 2015. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365* (2015).
- [276] Hu Yu, Jie Huang, Kaiwen Zheng, and Feng Zhao. 2023. High-quality image dehazing with diffusion model. *arXiv preprint arXiv:2308.11949* (2023).
- [277] Ye Yuan, Jiaming Song, Umar Iqbal, Arash Vahdat, and Jan Kautz. 2023. Physdiff: Physics-guided human motion diffusion model. In *Proceedings of the IEEE/CVF international conference on computer vision*. 16010–16021.
- [278] Sergey Zagoruyko and Nikos Komodakis. 2016. Wide Residual Networks. In *British Machine Vision Conference*.
- [279] Hong Zhang, Ying Liu, and Romit Maulik. 2025. Semi-implicit neural ordinary differential equations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 39. 22416–22424.
- [280] Jianxin Zhang, Josh Viktorov, Doosan Jung, and Emily Pitler. 2024. Efficient training of neural stochastic differential equations by matching finite dimensional distributions. In *International Conference on Learning Representations*.
- [281] Jianxin Zhang, Josh Viktorov, Doosan Jung, and Emily Pitler. 2025. Efficient Training of Neural Stochastic Differential Equations by Matching Finite Dimensional Distributions. In *International Conference on Learning Representations*.
- [282] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. 2017. Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising. *IEEE Transactions on Image Processing* 26, 7 (2017), 3142–3155. doi:10.1109/TIP.2017.2662206
- [283] Linan Zhang and Hayden Schaeffer. 2020. Forward stability of ResNet and its variants. *Journal of Mathematical Imaging and Vision* 62 (2020), 328–351.
- [284] Qinsheng Zhang and Yongxin Chen. 2021. Diffusion normalizing flow. In *Advances in Neural Information Processing Systems*, Vol. 34. 16280–16291.
- [285] Qinsheng Zhang and Yongxin Chen. 2022. Fast Sampling of Diffusion Models with Exponential Integrator. In *NeurIPS 2022 Workshop on Score-Based Methods*.
- [286] Qinsheng Zhang, Molei Tao, and Yongxin Chen. 2023. gDDIM: Generalized denoising diffusion implicit models. In *International Conference on Learning Representations*.
- [287] Xingcheng Zhang, Zhizhong Li, Chen Change Loy, and Dahua Lin. 2017. Polynet: A pursuit of structural diversity in very deep networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 718–726.
- [288] Yichi Zhang, Yici Yan, Alex Schwing, and Zhizhen Zhao. 2025. Hierarchical Rectified Flow Matching with Mini-Batch Couplings. *arXiv preprint arXiv:2507.13350* (2025).
- [289] Yichi Zhang, Yici Yan, Alex Schwing, and Zhizhen Zhao. 2025. Towards Hierarchical Rectified Flow. In *International Conference on Learning Representations*.
- [290] Hongjue Zhao, Yuchen Wang, Hairong Qi, Zijie Huang, Han Zhao, Lui Sha, and Huajie Shao. 2025. Accelerating Neural ODEs: A Variational Formulation-based Approach. In *International Conference on Learning Representations*.
- [291] Wenqing Zheng, Jiyang Xie, Xian Sun, and Zhanyu Ma. 2022. Structured Dropconnect for Uncertainty Inference in Image Classification. In *2022 IEEE International Conference on Image Processing (ICIP)*. IEEE, 366–370.
- [292] Wangchunshu Zhou, Tao Ge, Furu Wei, Ming Zhou, and Ke Xu. 2020. Scheduled DropHead: A Regularization Method for Transformer Models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*. 1971–1980.
- [293] Zhenyu Zhou, Defang Chen, Can Wang, and Chun Chen. 2024. Fast ode-based sampling for diffusion models in around 5 steps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 7777–7786.
- [294] Mai Zhu, Bo Chang, and Chong Fu. 2023. Convolutional neural networks combined with Runge–Kutta methods. *Neural Computing and Applications* 35, 2 (2023), 1629–1643.
- [295] Yuanzhi Zhu, Xingchao Liu, and Qiang Liu. 2024. SlimFlow: Training Smaller One-Step Diffusion Models with Rectified Flow. In *European Conference on Computer Vision*. 342–359.
- [296] Yixuan Zhu, Wenliang Zhao, Ao Li, Yansong Tang, Jie Zhou, and Jiwen Lu. 2024. FlowIE: Efficient Image Enhancement via Rectified Flow. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 13–22.